SBBD 2018

August 25 and 26 • Rio de Janeiro - Brazil

33rd Brazilian Symposium on DATABASES

Proceedings

Realização: SBC Sociedade Brasileira de Computação

Organização: IME

Apoio: CNPq Conselho Nacional de Desenvolvimento Científico e Tecnológico; CAPES

Patrocínio: Google

Apoio Institucional: PUC RIO

XXXIII BRAZILIAN SYMPOSIUM ON DATABASES (SBBD 2018)

August 25-26, 2018
Rio de Janeiro, RJ, Brazil

PROCEEDINGS Sociedade Brasileira de Computação (SBC)

PROMOTION
Sociedade Brasileira de Computação (SBC)

ORGANIZATION
Instituto Militar de Engenharia (IME)

SUPPORT
Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES)
Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq)

SPONSOR
Google

ACADEMIC SUPPORT
Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio)

005.74 Brazilian Symposium on Databases (SBBD 2018) (25.: 2018: Rio de Janeiro, RJ)

B827 Proceedings of 33nd Brazilan Symposium on Databases (SBBD 2018): August 25-26, 2018 – Rio de Janeiro, RJ, Brazil; organizadores: Bernadette Farias Lóscio; Carina Friedrich Dorneles; Maria Camila Nardini Barioni – Rio de Janeiro: SBC, 2018.

ISSN: 2016-5170
volume 01
316p.
Modo de acesso: http://sbbd.org.br/2018/

1. Computação - Congressos. 2. Bases de Dados – Congressos. l. Lóscio, Bernadette Farias. II. Dorneles, Carina Fiedrich. III. Barioni, Maria Camila Nardini. IV. Sociedade Brasileira de Computação. V. Título.

Message from the Local Organization Committee Chair

Welcome to the 33rd Brazilian Symposium on Databases! The Brazilian Symposium on Databases is the official database event of the Brazilian Computer Society (SBC) and the largest venue in Latin America for presentation and discussion of research results in the database domain. The 33rd edition of the symposium (SBBD 2018) was held in Rio de Janeiro, RJ, from August 25th to 26th, 2018. The local organization was performed by Instituto Militar de Engenharia (IME).

This year, for the first time, SBBD was held right before and at the same location of the 44thInternational Conference on Very Large Databases Conference (VLDB 2018), one of the main international conferences in the database area. This was a great opportunity for gathering national and international database researchers.

The SBBD 2018 program offered a variety of activities, suited for an audience ranging from undergraduate to Ph.D. students, database professionals, practitioners and researchers. The Program included: 4 invited talks and 2 tutorials, presented by distinguished speakers from Brazil, USA and Switzerland; 6 technical sessions; demos and applications session; posters sessions; thesis and dissertations workshop.

The excellence of SBBD 2018 program is the result of the competence and effort of a large community, which we gratefully acknowledge. The various sections of these proceedings list in detail those that contributed to the SBBD 2018 edition. We thank the symposium chairs and our colleagues of the local organization committee who donated their precious time to make SBBD 2018 a reality. We specially thank IME and its Post-graduation Programs (PGSC and PGED), which allowed their staff and students to help on the many tasks of the event preparation. We are also grateful to the SBC board for their support and to the steering committee members for their help, advice and support. Further, we thank the program committee members and external reviewers for the high quality reviews, and the authors who submitted their papers to SBBD 2018. Finally, we are grateful to our sponsors. Without their support we would not be able to organize this annual event that brings together our community.

We had a great time hosting SBBD 2018 in Rio de Janeiro!

Maria Claudia Reis Cavalcanti (IME)
Local Organization Chair – SBBD 2018

Foreword

It is a great pleasure to introduce the Proceedings of the Brazilian Symposium on Databases (SBBD) with the full, short and industrial papers accepted for presentation at the 33rd edition of the symposium. SBBD 2018 was held in Rio de Janeiro, in the state of Rio de Janeiro, Brazil, from August 25th to August 26th, 2018. It was organized by Seção de Engenharia de Computação, Instituto Militar de Engenharia (IME). This edition was held in two days of August instead of the traditional four days of October to motivate and to facilitate the participation of the database Brazilian community in the VLDB 2018 (44th International Conference on Very Large Data Bases). VLDB 2018 also took place in Rio de Janeiro, Brazil, from August 27th to August 31st.

SBBD is the official database event of the Brazilian Computer Society (SBC) and the largest venue in Latin America for presentation and discussion of research results in the databases domain. Along with technical sessions, SBBD includes invited talks and tutorials given by distinguished speakers from the national and international research communities. SBBD regularly promotes a demos and applications session, and a thesis and dissertations workshop as co-located events.

All papers presented in technical sessions during the event reported interesting results or proposed novel thought-provoking ideas in several subjects on the databases and related areas. For the 2018 edition, SBBD accepted six categories of submissions: JIDM articles, full papers, short papers, industrial papers, vision papers, and distinguished published papers.

Submissions to the JIDM category could be made throughout the year. The review process was conducted by the editorial board of JIDM, leaded by the editor-in-chief Angelo Brayner. This year, no JIDM articles were presented during the event. The Full papers track had one cycle of submissions with deadline in March. The review process for the SBBD full papers was performed in one round with a rebuttal phase. Authors were initially notified with the reviews and had a few days for answering the reviewers' comments during the rebuttal phase. After evaluating the rebuttal comments during the discussion period, a final decision was achieved. Out of 40 submitted papers and 17 were accepted as full papers(acceptance rate of 42,5%).

Short, Industrial and Vision papers track had a single cycle of submissions with deadline in June. There was no rebuttal phase. This year's Short, Industrial and Vision track features 15 short papers and 2 industrial papers. Initially, there were 36 submissions in the track, 29 in the Short category and 7 in the Industrial category. The best full and short papers received award certificates during the event and will be invited to submit extended versions of their papers to JIDM.

Distinguished Published Papers is a category of submission introduced in SBBD 2017. It aimed to attract the best papers of the Brazilian community, published or accepted for publication by a first-class database conference, and give the authors the opportunity to present their work during the event. There was one submission that was accepted by the SBBD Steering Committee, on the grounds of relevance of the original publication venue to the database community.

The topics with more submissions among full and short papers (according to the author's selection from the Topics of Interest) were: Data Analytics and Data Visualization (17 submissions), Data on the Web (16 submissions), Algorithms and Techniques for Data Mining (14 submissions), Semantic Web, Linked Data and Ontologies (14 submissions), Data Modeling (11 submissions), Information Integration and Interoperability (9 submissions), Social Data Processing (7 submissions), Graph Databases (7 submissions) and Authorization, Privacy and Security in Databases (7 submissions).

The Proceedings of SBBD are the result of the collective effort of a large community, which we gratefully acknowledge. We thank the SBBD 2018 local organization committee and its symposium chairs, who worked hard to guarantee an outstanding event. We do not have enough words to thank all committee members and external reviewers for their commitment and high quality reviews. We are also grateful to the steering committee members for their help, advice and support. Finally, we are grateful to the authors who submitted their work to SBBD 2018.

Bernadette F. Lóscio (UFPE)
Program Chair – SBBD 2018 – Full Papers

Carina F. Dorneles (UFSC)
Program Chair – SBBD 2018 – Short, Vision and Industrial Papers

Editorial for the Tutorials Track

Tutorials at the Brazilian Symposium on Databases (SBBD) have the goal to present introductory and advanced discussions on topics within the area of databases. Introductory tutorials target an audience consisting of advanced undergraduate and graduate students, as well as attendees from industry. Advanced tutorials, on the other hand, cover a state-of-the-art topic, motivating and exposing potential research paths.

The two accepted tutorials this year are related to relevant nowadays topics. The first one is entitled "In-Memory Analytic DBMSs: Design and Lessons Learned" and will discuss how in-memory analytic DBMSs are designed and built and outline the architecture of some state-of-art in-memory database systems, stressing the characteristics that differentiate them from the traditional DBMS design literature. It will be presented by Pedro Eugenio Rocha Pedreira, Software Engineer at Facebook focused on database research.

The second tutorial, "Coleta, Integração e Pré-processamento de Dados de Múltiplas Fontes", will be presented by Natércia Aguilar (MSc. student), Michele A. Brandão (Postdoctoral researcher) and Mirella M. Moro (Associate Professor) from Federal University of Minas Gerais (UFMG). They will talk about the main challenges related to collecting, integrating and preprocessing data from multiple Web sources. This tutorial will address these three issues in an integrated way with a focus on practical and research questions.

This year we had four excellent submissions, and I would like to thank all the authors of submitted proposals. I am also grateful to the tutorials program committee members for the high-quality reviews. Also, I would like to invite all of you to attend and take advantage of the selected tutorials.

Maria Camila Nardini Barioni (UFU)
Program Chair – SBBD 2018 – Tutorials

Full Papers – Technical Committee

Program Chair

Bernadette F. Lóscio (UFPE)

Program Committee

Alexandre Plastino (UFF)
Altigran Soares da Silva (UFAM)
Ana Carolina Salgado (UFPE)
André Santanchè (UNICAMP)
Angelo Brayner (UFC)
Caetano Traina Jr (ICMC/USP)
Carina F. Dorneles (UFSC)
Carmem Hara (UFPR)
Celso Hirata (ITA)
Clodoveu Davis (UFMG)
Cristina Ciferri (USP)
Damires Souza (IFPB)
Daniel de Oliveira (UFF)
Daniel Kaster (UEL)
Denio Duarte (UFFS)
Divesh Srivastava (AT&T Labs-Research, EUA)
Duncan Ruiz (PUCRS)
Edleno Moura (UFAM)
Eduardo Ogasawara (CEFET/RJ)
Elaine Sousa (USP)
Fernanda Baião (UNIRIO)
Genoveva Vargas-Solar (CNRS, França)
Humberto Razente (UFU)
Javam Machado (UFC)
João Eduardo Ferreira (IME/USP)
Jonice de Oliveira Sampaio Oliveira (IM/UFRJ)
José Palazzo Moreira de Oliveira (UFRGS)
Karin Becker (UFRGS)
Kelly Braghetto (IME/USP)
Khalid Belhajjame (U. Paris Dauphine, França)
Luciano Barbosa (UFPE)
Marco Antonio Casanova (PUC-Rio)
Marcos Gonçalves (UFMG)
Maria Camila Nardini Barioni (UFU)
Mirella Moro (UFMG)
Mirian Halfeld-Ferrari (U. d'Orleans, França)
Pedro Eugenio Rocha Pedreira (Facebook Inc.)
Renata Galante (UFRGS)
Renato Fileto (UFSC)
Ricardo Ciferri (UFSCar)
Ricardo Torres (UNICAMP)
Ronaldo Mello (UFSC)
Sergio Lifschitz (PUC-Rio)
Valéria C. Times (UFPE)
Vanessa Braganholo (UFF)
Vania Bogorny (UFSC)
Vaninha Vieira (UFBA)
Wagner Meira Jr. (UFMG)
Zoubida Kedad (UVSQ, França)

Additional Reviewers

Anderson Chaves Carniel (USP)
Christian Quevedo (PUCRS)
Diego Pessoa (UFPE)
Jonnathan Carvalho (IFFluminense)
Levy Souza (UFMG)
Marcelo Iury S. Oliveira (UFRPE)
Mauro Roisenberg (UFSC)
Nielsen Machado (PUCRS)
Pablo Silva (UFF)
Pedro Losco Takecian (IME/USP)
Priscilla Vieira (UFPB)

Short, Vision and Industrial Papers – Technical Committee

Program Chair

Carina F. Dorneles (UFSC)

Program Committee

Alessandreia Oliveira (UFJF)
Altigran Soares da Silva (UFAM)
Ana Carolina Almeida (UERJ)
Anderson Ferreira (UFOP)
Angelo Brayner (UFC)
Bernadette F. Lóscio (UFPE)
Carlos Eduardo Pires (UFCG)
Carmem Hara (UFPR)
Cristina Ciferri (USP)
Damires Souza (IFPB)
Daniel de Oliveira (UFF)
Daniel Kaster (UEL)
Daniela Barreiro Claro (UFBA)
Deise Saccol (UFSM)
Denio Duarte (UFFS)
Duncan Ruiz (PUCRS)
Eduardo Borges (FURG)
Eduardo de Almeida (UFPR)
Eduardo Ogasawara (CEFET/RJ)
Elaine Sousa (USP)
Eveline Sacramento (FUNCEME)
Fernanda Baião (UNIRIO)
Flávio R. C. Sousa (UFC)
Helena Ribeiro (UCS)
Humberto Razente (UFU)
João Eduardo Ferreira (IME/USP)
Jonas Dias (DELL EMC)
Jonice de Oliveira Sampaio Oliveira (IM/UFRJ)
José Antonio Macêdo (UFC)
José de Aguiar Moraes Filho (UNIFOR)
José Monteiro (UFC)
José Palazzo Moreira de Oliveira (UFRGS)
Karin Becker (UFRGS)
Kelly Braghetto (IME/USP)
Luciano Barbosa (UFPE)
Luiz Celso Gomes Jr (UTFPR)
Luiz Manoel Rocha Gadelha Júnior (LNCC)
Maria Camila Nardini Barioni (UFU)
Maristela Holanda (UnB)
Michele Brandão (UFMG)
Mirella Moro (UFMG)
Moisés Carvalho (UFAM)
Pedro Eugenio Rocha Pedreira (Facebook Inc.)
Raquel Stasiu (PUCPR / UTFPR)
Raqueline Penteado (UEM)
Rebeca Schroeder (UDESC)
Renata Galante (UFRGS)
Renato Fileto (UFSC)
Robson Cordeiro (USP)
Robson Fidalgo (UFPE)
Sergio Lifschitz (PUC-Rio)
Sergio Mergen (UFSM)
Thiago Henrique Silva (UTFPR)
Ticiana Coelho da Silva (UFC)
Valéria C. Times (UFPE)
Vanessa Braganholo (UFF)
Vaninha Vieira (UFBA)

Additional Reviewers

Marcelo Iury S. Oliveira (UFRPE)
Eduardo Pena (UFPR)
Guilherme Queiroz Vasconcelos (USP)
Nielsen Luiz Rechia Machado (PUCRS)

Tutorials – Technical Committee

Program Chair

Maria Camila Nardini Barioni (UFU)

Program Committee

Agma Juci Machado Traina (ICMC/USP)
Ana Carolina Salgado (UFPE)
Caetano Traina Jr (ICMC/USP)
Javam Machado (UFC)
Marta Mattoso (COPPE/UFRJ)

Table of Contents

Full Papers

Performance Evaluation and Benchmarking and; Concurrency Control and Recovery

Short, Vision and Industrial Papers

Tutorials

Keynotes

Author Index 315

SBBD 2018

Full Papers

WANQA: Uma Abordagem para Identificar Novas Questões Não Respondíveis em Comunidades de Perguntas e Respostas

Lucas V. Knochenhauer[1], **Carina F. Dorneles**[1], **Leandro K. Wives** [2]

[1]PPGCC / INE - Universidade Federal de Santa Catarina (UFSC)
Florianópolis – SC – Brasil

[2]PPGC / INF - Universidade Federal do Rio Grande do Sul (UFRGS)
Porto Alegre – RS – Brasil

{luckasx}@gmail.com, {dorneles}@gmail.com, {wives}@inf.ufrgs.br

Abstract. *Big knowledge repositories are on the web and Question and Answer Communities (CQAs) are one of the most collaborative. Daily, their users post a large volume of questions and a great part of them receives no answers, becoming it useless content. Previous works that aim to solve this problem are dependent on the given characteristics of each community. This article proposes an approach based on a classification that results a model able to classify whether a new question is answerable or not. It uses features available in most CQAs. Experiments with data from different CQAs show that the proposal fulfills its goals.*

Resumo. *Grandes repositórios de conhecimento estão distribuídos pela Web, sendo que um dos mais colaborativos são as comunidades de perguntas e respostas (CQAs). Diariamente, os seus usuários postam grandes volumes de questões e boa parte delas não recebe respostas, tornando-se conteúdo inútil. Trabalhos existentes, que se propõem a resolver esse problema, são dependentes das características presentes em cada comunidade. Neste artigo, é proposta uma abordagem baseada em classificação, que gera um modelo capaz de identificar uma nova questão como respondível ou não, usando características presentes na grande maioria das CQAs. Experimentos com dados de diferentes CQAs mostram que o método proposto cumpre seus objetivos.*

1. Introdução

A disponibilização de informação na Web pode ser feita de várias formas, inclusive através de comunidades de usuários. Nas comunidades de perguntas e respostas, por exemplo, chamadas de *Community Question Answering* (CQA) [Srba and Bielikova 2016], os usuários interagem entre si, postando questões e soluções para determinados assuntos. A interação entre os membros produz diariamente uma grande quantidade de informação e conhecimento sobre diferentes temas. A Figura 1 apresenta um exemplo de questão postada e respostas dadas pelos usuários.

Nessas comunidades, no entanto, um dos problemas é a falta de respostas para determinadas questões, e questões sem respostas não são boas para as CQAs pois acumulam conteúdo inútil para seus usuários. Para ajudar a minimizar essa deficiência, algumas abordagens foram propostas na literatura [Yang et al. 2011a, Dror et al. 2013,

Asaduzzaman et al. 2013, Fong et al. 2015] para determinar se uma nova questão está propensa a receber respostas ou não. Isso porque caso uma questão seja reconhecida como não respondível no momento de sua postagem, isso pode levar o usuário a rever o texto do seu questionamento antes dela ser disponibilizada publicamente. Com isso, aqueles que desejam responder à dúvida apresentada terão menos dificuldades no entendimento, o que facilita a resolução do questionamento.

Figura 1. Exemplo de Perguntas e Respostas

Na literatura, diversos trabalhos [Yang et al. 2011a, Dror et al. 2013, Asaduzzaman et al. 2013, Fong et al. 2015] foram propostos com o intuito de classificar novas questões como respondíveis ou não. De forma geral, a maioria deles faz uso de um conjunto de características próprias de cada CQA. Apesar de otimizar a tarefa, isso gera a necessidade de adaptações em outros contextos, visto que cada comunidade tem sua própria estrutura e forma de interação. Nesses conjuntos não há como inferir quais características melhor identificam as questões. Em algumas das abordagens, por exemplo, ignoraram-se os termos usados nos textos das questões, cuja presença pode revelar se a pergunta está adequada para ser respondida ou não.

Neste artigo, propõe-se um método para classificar uma nova questão como respondível ou não, usando características comumente presentes nas comunidades, independente de suas particularidades, criando-se uma abordagem mais genérica. Porém, a generalidade da proposta pode acarretar diminuição da acurácia em algumas CQAs. Diante disso, para mitigar esse problema, propõe-se a ponderação das características de uma questão, de acordo com a sua importância em uma dada categoria de questão. Por exemplo, a característica "há código de programação" deve ter mais peso para perguntas da categoria "programação" do que para aquelas da categoria de "esportes". Com esse método, é possível treinar um modelo de classificador para cada uma das categorias presentes na CQA e aplicá-lo às postagens dos usuários. As contribuições do presente trabalho podem ser sumarizadas da seguinte forma: (i) definição das características comuns à maioria das comunidades, provenientes das novas questões e dos usuários que as postam; (ii) uso da atribuição de pesos às características a fim de ressaltar sua influência no recebimento de respostas; e (iii) experimentos que confirmam a melhoria da acurácia da classificação com o uso de características comuns ponderadas.

A proposta é validada com experimentos realizados sobre fontes de dados advindos de diferentes CQAs de forma a se comparar a eficácia da proposta e a variação nos resultados com a aplicação de diferentes algoritmos de classificação e de atribuição e variação de pesos. Os conjuntos de dados foram obtidos das comu-

nidades *StackOverflow*[1], *Mathematics*[2] e *Cross Validated* [3]. A mensuração dos resultados foi feita usando as métricas de acurácia, revocação, precisão e medida F [Baeza-Yates and Ribeiro-Neto 2008]. Ao comparar os resultados, pode-se ver que a proposta traz melhorias na classificação em relação aos três *baselines* estabelecidos.

Este artigo está organizado como segue. A Seção 2 detalha a proposta, descrevendo a extração, seleção das características, e o treinamento do modelo. Em seguida, na Seção 3 apresentam-se a configuração dos experimentos executados e os respectivos resultados. Na Seção 4, são discutidos os trabalhos relacionados. Finalmente, na Seção 5 são apresentados trabalhos futuros e conclusões.

2. WANQA - Weighting Analysis for New Question Answerability

O processo de classificação envolve três passos principais: (i) Extração de Características; (ii) Seleção das Características; e (iii) Treinamento do Modelo. A visão geral da proposta é apresentada na Figura 2. Como entrada, são utilizados os dados das novas questões e dos usuários que as postaram. No primeiro passo, é feito um levantamento das características disponíveis nas questões da entrada, incluindo a criação de um vetor *tf-idf*. Logo em seguida, faz-se a atribuição de pesos às características vindas da etapa anterior e filtra-se um subconjunto das características com maiores pesos. No último passo é realizado o treinamento com um algoritmo de classificação, o qual gera um modelo treinado.

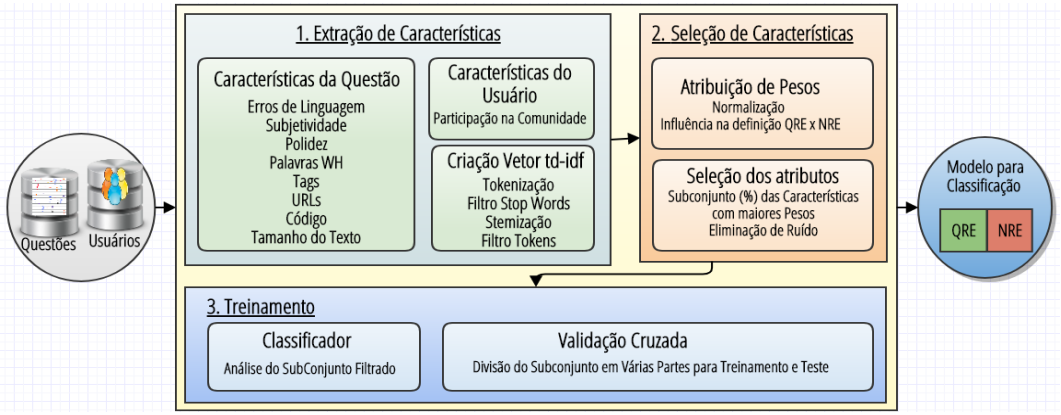


Figura 2. Fluxo Criação do Modelo para Classificação de Questões

A fase de treinamento é realizada através de algoritmos de classificação. Para tanto, foram criadas duas classes: *Questão Respondível - QRE* e *Questão Não Respondível - NRE*. As questões são consideradas respondíveis quando possuem características semelhantes às questões já respondidas na comunidade e na categoria em que esta foi marcada. A classificação das questões não respondíveis funciona do mesmo modo. Através do histórico das questões elenca-se as características que não trazem respostas. Cada categoria possui suas características mais relevantes tanto para o recebimento quanto a falta de respostas. Um exemplo simplificado seria: Questões sobre SQL com termos envolvendo *stored procedures* recebem mais respostas do que Questões sobre SQL com código *javascript*.

[1] `http://www.stackoverflow.com` (Programação)
[2] `http://math.stackexchange.com/` (Matemática)
[3] `http://stats.stackexchange.com/` (Estatística e Análise de Dados)

Nesta abordagem, é empregada a validação cruzada do tipo *k-fold*. Essa validação divide o conjunto de dados em *k* partes das quais $k-1$ são usadas para treinar o algoritmo classificador e a parte restante é usada para testes do modelo treinado, onde o número de classificações corretas e incorretas é medido. O processo é repetido k-1 vezes e uma média dos valores é utilizada para aferir o modelo. Após esse treinamento são gerados o modelo treinado e os resultados contendo o número de classificações corretas e incorretas. O modelo resultante é o que será efetivamente aplicado nas novas questões postadas para classificá-las em QRE ou NRE.

2.1. Extração de Características

Nesta primeira etapa são extraídas características presentes nas novas questões e na descrição do perfil dos usuários. A escolha das características a serem usadas foi feita com base na análise das características consideradas nos trabalhos relacionados e levando em conta os dados disponíveis desde o momento da postagem. Além disso, para que a abordagem não seja dependente de uma comunidade específica, foram analisadas características disponíveis na grande maioria das comunidades.

As características extraídas dos perfis dos usuários são as seguintes: número de dias como membro, número de questões postadas pelo usuário, número de respostas postadas pelo usuário e proporção de respostas em relação à quantidade de perguntas postadas. Para extração do conjunto de características das questões, é necessário obter Título, Descrição, Data e Categorias de cada uma delas.

A Tabela 1 apresenta as 16 características extraídas das novas questões. O Tamanho do Título de uma questão é relevante, pois resume o questionamento em poucas frases. Quando o título é muito vago ele não explica o real problema que se está lidando. Uma característica identificada como relevante é "Título inicia com palavra WH" é uma característica cujo valor é booleano. Isso porque as palavras WH (*what, why, when, who, which, how, whose, whom*) são uma forma de especificar o que deve ser respondido. A presença ou não dessas palavras interrogativas pode ser compreendida como uma forma de medir a objetividade ou subjetividade da pergunta. Erros de Linguagem indicam se os textos estão mal escritos, podendo dificultar o entendimento do problema apresentado e por consequência serem ignorados pelos demais usuários. A descrição de uma questão detalha o problema. Em algumas comunidades esse texto é opcional, a adição dessa informação depende do contexto e da dificuldade da pergunta. A característica "Há Código na Descrição" é um valor booleano indicando se há código-fonte na questão. Geralmente, nos assuntos de programação os problemas estão relacionados aos códigos de algum *software*. Essa característica visa identificar se a inclusão de um código é relevante ou não para a categoria da questão sendo classificada. Seno e Cosseno do Dia e da Hora são características calculadas pelas fórmulas $seno(2\pi * dia/7)$, $cosseno(2\pi * dia/7)$, $seno(2\pi * hora/24)$ e $cosseno(2\pi * hora/24)$. Esses cálculos separam as propriedades Dia da Semana e Hora da Postagem em duas dimensões, positiva e negativa, facilitando a separação de forma linear para os algoritmos de classificação [Zhou and Fong 2016].

Após a extração das características, cria-se um vetor *tf-idf* [Aggarwal 2015] com todos os termos encontrados nas questões e seus respectivos pesos. A determinação do valor *tf-idf* visa identificar os termos que são relevantes para a categoria das questões (Futebol, *Javascript*, etc.) que estão sendo usadas para a criação do modelo. Ressalta-se

que antes da atribuição dos valores é feito um pré-processamento removendo os termos muito comuns, *stop words*, além de *stemização* para reduzir a quantidade de vocábulos. A Figura 3 apresenta um exemplo de valores *tf-idf* atribuídos a termos presentes em algumas questões. Quanto maior o valor atribuído, significa que o termo aparece menos vezes no conjunto de questões em relação aos demais.

Tabela 1. Características extraídas das novas questões

Característica	**Descrição**
Erros de linguagem	Quantidade de erros gramaticais encontrados na descrição da questão.
Tamanho do Título/Descrição	Quantidade de caracteres no Título/Descrição.
Legibilidade	Facilidade/dificuldade de entendimento do texto apresentado. Valor calculado com base no consenso do cálculo de várias fórmulas da literatura.
Subjetividade	Valor numérico indicando se a postagem busca soluções ou opiniões, pode receber valor no intervalo de 0 (muito objetiva) a 1 (muito subjetiva).
Código na Descrição	Valor booleano indicando se há código de programação na questão.
Polidez	Quantidade de palavras de gentileza *(thank, thanks, please, could, would, help)* utilizadas na descrição.
Seno e Cosseno do Dia	Valores calculados com base na propriedade Dia da Semana.
Seno e Cosseno da Hora	Valores calculados com base na Hora da Postagem. Visam identificar se os horários das postagens influenciam no recebimento de respostas.
Contagem de *tags*	Quantidade de categorias em que a questão foi marcada. Identifica a abrangência da questão em relação aos usuários solucionadores.
Título inicia com palavra WH	Valor booleano que indica se o título da questão inicia com "WH".
Contagem de URL	Quantidade de URLs informadas na descrição. Esse valor mostra se houve uma pesquisa prévia na tentativa de resolver o problema.
Contagem de palavras WH	Quantidade de palavras que iniciam com "WH" no Título/Descrição.

2.2. Seleção de Características

Como o tamanho do conjunto gerado na etapa anterior pode ser muito grande, devido à criação do vetor *tf-idf*, é necessária a definição de um subconjunto das características mais relevantes para uso na fase de treinamento. A relevância é dada pela atribuição de pesos que relaciona as características com as classes *QRE* e *NRE*. Quanto maior o peso atribuído, maior a relevância da característica na classificação. Dessa forma, na segunda etapa do processo, aplica-se o método de *Feature Selection* baseada em filtros [Aggarwal 2015] para detecção das características mais relevantes geradas anteriormente. A Figura 4 apresenta um exemplo de pesos atribuídos a um conjunto de características.

Question	actual	default	known	return	set	string
1	0	0	0,193907	0	0	0
2	0	0	0	0,205024	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0,062495	0	0	0,197223	0,037924	0,05981
6	0,058666	0	0	0	0	0,028073
7	0	0,052391	0	0	0,1026	0
8	0	0	0	0	0	0
9	0	0	0	0	0	0
10	0,046418	0	0	0	0,084505	0

Figura 3. Exemplo de valores tf-idf atríbuídos

attribute	weight
QuestionLength	0.257
tagsCount	0.214
bodyHasCode	0.191
android	0.168
LanguageErrors	0.162
titleStartsWithWH	0.152

Figura 4. Exemplo de pesos atríbuídos

A seleção de características baseada em filtros inicia com a atribuição de pesos. Essa atribuição indica um valor que determina o grau de relacionamento da característica com as classes das questões *QRE* e *NRE*. Há diferentes formas para calcular esses pesos. Neste trabalho, foram testadas 3 técnicas diferentes *Information Gain*, *Correlação*

de Pearson e *GINI Index* [Aggarwal 2015], cuja eficácia é apresentada e comparada experimentalmente mais adiante.

Como a produção dos modelos é feita para cada categoria de questão, os treinamentos são realizados individualmente para cada categoria, e as características têm diferentes pesos em cada treinamento. Por exemplo, a característica "Há código na descrição" pode ter um peso maior para as perguntas que envolvem linguagens de programação, mas para dúvidas relacionadas à esportes pode ser irrelevante. Essa estratégia visa reduzir eventuais ruídos que as características irrelevantes podem causar nos modelos. O tamanho do subconjunto com menos ruídos não é conhecido antes do treinamento, por isso é necessário definir nesta etapa quantas das características com melhores pesos são usadas para a criação do modelo. A seleção de um subconjunto tem por objetivo aumentar a acurácia na classificação.

3. Experimentos

Para avaliar a abordagem proposta, foram realizados experimentos usando as características apresentadas na Seção 2, bem como a atribuição de pesos para definir a relevância de cada uma delas. Para fins de análise, a proposta foi comparadas a outras estratégias, gerando comparativos com os seguintes *baselines*: (B1) implementação da proposta de Fong, Zhou e Moutinho [Fong et al. 2015], que usa uma técnica de *swarm* para otimizar a acurácia treinando vários subconjuntos; (B2) conjunto de características sem a criação do vetor *tf-idf*; e (B3) resultados obtidos com o vetor *tf-idf* sem aplicar *feature selection*. O *baseline* B3 equivale às estratégias adotadas por Yang et al. [Yang et al. 2011a] e Dror, Maarek e Szpektor [Dror et al. 2013], onde foram usados os termos encontrados nas questões para, respectivamente, a criação de tópicos e análise da presença desses termos nos textos.

3.1. Conjuntos de Dados

Os experimentos foram executados com 3 conjuntos de questões, cada um deles referente a uma categoria de uma comunidade: categoria "Java", da comunidade *StackOverflow*; (ii) categoria "Álgebra Linear", da comunidade *Mathematics*; e (iii) categoria "Regressão" da comunidade *Cross Validated*. As CQAs citadas pertencem ao mesmo grupo, *StackExchange*[4], porém, os assuntos tratados, características e os membros são distintos uns dos outros. Os conjuntos de questões foram montados de forma que as classes QRE e NRE tivessem um número semelhante de registros. A quantidade de questões empregadas nos experimentos em cada categoria foi de 3000 sobre Java, 2000 de Regressão e 2000 de Álgebra Linear. Na prática, há muito mais questões respondidas do que o contrário, atualmente a distribuição das perguntas sem respostas está como demonstrado na Tabela 2. Quando os dados são desbalanceados os algoritmos classificadores tendem para a classe sobressalente. Sendo assim, optou-se pelo balanceamento através de amostragem com o objetivo de dar equilíbrio aos atributos das questões não respondidas.

3.2. Descrição da avaliação

Os principais objetivos para execução dos experimentos são: (i) realizar uma análise comparativa entre as diferentes formas de classificação (considerando os *baselines* apresentados); (ii) analisar o efeito do uso de características comuns em CQAs; e (iii) investigar a

[4] `http://stackexchange.com/`

Tabela 2. Distribuição das Questões nas Categorias Experimentadas

Comunidade	Categoria	# Questões	# Sem Respostas	% Sem Respostas
Stackoverflow	Java	1.300.000	164.000	12,6%
Mathematics	Álgebra Linear	70.300	9.600	13,6%
Cross Validated	Regressão	15.500	4.800	30,9%

consequência do uso de peso em caraterísticas extraídas das questões, dada a categoria na qual ela foi associada. As métricas de avaliação utilizadas são acurácia (A), precisão (P), revocação (R) e medida-F (F) para as duas classes possíveis, QRE e NRE. O cálculo das métricas é feito com as equações apresentadas na Figura 5, onde TP, TN, FP e FN significam respectivamente verdadeiros positivos e negativos, e falsos positivos e negativos.

$$A = \frac{TP+TN}{TP+FP+TN+FN} \quad P = \frac{TP}{TP+FP} \quad R = \frac{TP}{TP+FN} \quad F = 2 * \frac{P*R}{P+R}$$

Figura 5. Métricas de Avaliação

Na etapa de seleção de características, foram aplicados três métodos: Correlação de *Pearson* (PC), *Information Gain* (IG) e *GINI Index* (GI). A intenção é analisar o relacionamento das características com as classes QRE e NRE e a acurácia resultante conforme cada método de peso empregado. Na etapa de treinamento, foram usados os classificadores disponíveis na ferramenta *Rapidminer Studio*[5]: SVM, Naive Bayes (NB), *Hyperpipes* (HP) e Regressão Logística (RL). Para evitar *overfitting*, ou seja, que o classificador não se ajuste demais ao conjunto de questões do treinamento, foi utilizada a técnica *k-fold* de validação cruzada, sendo k = 10. Dessa forma, o algoritmo classificador executa 10 rodadas dividindo os dados em 10 partes (*folds*) sendo 9 para treinamento e 1 para validação. Para cada método de peso e classificador foram selecionados 9 subconjuntos variando de 10% a 90% das características com os melhores pesos atribuídos. Os resultados de cada um dos 9 subconjuntos têm por propósito verificar a quantidade de características necessárias para obtenção de melhor acurácia. Os dados, códigos e processos manipulados estão disponíveis em `http://github.com/Luckasx/NQClassificationExperiments/`.

3.3. Resultados

No total, foram executadas 360 configurações diferentes em busca da melhor acurácia. Os melhores resultados obtidos nos experimentos estão apresentados na Tabela 3. As linhas na tabela representam a configuração dos *baselines* (B1, B2, B3) e diferentes configurações da proposta apresentada (P.*x*) com os três métodos de peso testados - Correlação de *Pearson* (PC), *Information Gain* (IG) e *GINI Index* (GI) que resultaram em melhor acurácia. Além da acurácia, a tabela exibe a precisão e a revocação obtidas para as classes QRE e NRE nos 3 conjuntos de questões.

A proposta deste trabalho obtém melhor acurácia quando comparada aos demais *baselines*. Dentre os 3 métodos de pesos aplicados, a filtragem com Correlação de *Pearson* (PC) teve melhor acurácia. Pode-se perceber também que os subconjuntos de características nos experimentos P.*x* variam entre 30% e 70% do conjunto total. A acurácia da classificação sem a adição do vetor *tf-idf* (B2) gerou menor acurácia do que o contrário

[5]`http://rapidminer.com/products/studio/`

(B3). Entretanto, a simples adição do vetor *tf-idf* não é suficiente para obter melhora significativa na classificação. O filtro no conjunto total de características (experimentos P.*x*) obteve resultados melhores de 12,94 até 23,8 pontos percentuais do que mantendo todas as características. Isso evidencia que os filtros geram um subconjunto mais adequado à classificação das perguntas. O experimento do *baseline* B1 obteve melhor acurácia do que os experimentos de B2 e de B3. Porém, ressalta-se que os dados continham valores de características específicas às comunidades *StackExchange*, tais como o número de votos que as questões receberam e pontos de reputação dos usuários.

Tabela 3. Melhores Resultados de Acurácia na Classificação

Experimento	Algoritmo	Acurácia	Filtro	% Precisão		% Revocação		% Medida-F	
				QRE	NRE	QRE	NRE	QRE	NRE
Java									
B1	RL	83,20%	-	87,61	79,71	77,33	89,07	82,15	84,13
B2	RL	70,10%	-	69,74	70,47	71,00	69,20	70,36	69,83
B3	RL	74,03%	-	73,45	74,64	75,27	72,80	74,35	73,71
P.PC	**NB**	**86,97%**	**50%**	**83,14**	**91,79**	**92,73**	**81,20**	**87,67**	**86,17**
P.IG	NB	84,13%	70%	82,49	85,96	86,67	81,60	84,53	83,72
P.GI	NB	83,83%	70%	82,06	85,82	86,60	81,07	84,27	83,38
Álgebra Linear									
B1	RL	69,70%	-	69,35	70,06	70,60	68,80	69,97	69,42
B2	RL	64,05%	-	63,39	64,77	66,50	61,60	64,91	63,15
B3	RL	65,30%	-	65,00	65,61	66,30	64,30	65,64	64,95
P.PC	**NB**	**79,65%**	**40%**	**73,89**	**89,06**	**91,70**	**67,60**	**81,84%**	**76,86**
P.IG	SVM	75,30%	30%	73,73	77,09	78,60	72,00	76,09	74,46
P.GI	NB	74,95%	60%	71,13	80,46	84,00	65,90	77,03	72,46
Regressão									
B1	RL	61,50%	-	60,63	62,53	65,60	57,40	63,02	59,86
B2	RL	57,20%	-	56,95	57,47	59,00	55,40	57,96	56,42
B3	SVM	57,65%	-	57,10	58,29	61,50	53,80	59,22	55,96
P.PC	**NB**	**81,45%**	**40%**	**77,61**	**86,53**	**88,40**	**74,50**	**82,65**	**80,07**
P.IG	NB	78,55%	60%	76,76	80,60	81,90	75,20	79,25	77,81
P.GI	NB	77,60%	60%	75,79	79,68	81,10	74,10	78,36	76,79

Em relação à precisão, o método proposto é melhor do que os demais tanto para as QRE quanto para as NRE. Exceto no conjunto de perguntas Java, onde o *baseline* B1 obteve melhor precisão para as QRE. Os valores de Revocação comportam-se de maneira semelhante, mas P.*x* gerou melhor revocação para todas QRE, enquanto a configuração B1 foi melhor nas bases de questões NRE de Java e Álgebra Linear. Os valores da Medida-F ficaram todos próximos à acurácia resultante. Esses resultados mostram que o método proposto está classificando corretamente a maior parte das QRE e NRE.

Dentre os 4 algoritmos classificadores testados, Naive Bayes resultou em melhor acurácia. Todos os classificadores obtiveram melhores resultados em relação aos *baselines* B2 e B3. A melhor acurácia de SVM ficou muito próxima da alcançada por Naive Bayes. A diferença variou entre 0,15 e 3,67 pontos percentuais. *Hyperpipes* atingiu acurácia pouco menor. Apesar de ter revocação maior que 90% em relação às NRE, para a classe QRE alcançou em média 60%, prejudicando a acurácia final. A diferença em relação a Naive Bayes variou de 6,56 a 12,7 pontos percentuais. Os melhores resultados de Regressão Logística decorreram da classificação do menor subconjunto, com apenas 10% das características. Nesse caso, o menor subconjunto gerou melhores resultados por causa da regularização L1, uma maneira de evitar *overfitting* penalizando coeficien-

tes grandes. Os valores resultantes para acurácia foram de 7,9 a 13 pontos percentuais menores do que os obtidos com Naive Bayes. Sendo assim, a partir dos resultados dos experimentos pode-se recomendar o uso dos dois classificadores, Naive Bayes e SVM, de modo a obter melhor acurácia na classificação.

A Figura 6 apresenta os valores de acurácia obtidos na execução de P.PC, P.IG e P.GI para o classificador Naive Bayes em cada subconjunto conforme os pesos aplicados. Os gráficos demonstram que os subconjuntos derivados da Correlação de Pearson tiveram melhor acurácia nos 3 conjuntos de questões. Outro comportamento que se destaca é a melhora da classificação conforme se aumenta o tamanho do subconjunto. A partir do melhor ponto, por volta de 50% do conjunto total, há uma queda brusca na acurácia. Essa piora deve-se ao fato de que boa parte do conjunto torna-se ruído para o algoritmo classificador, ou seja, não foi considerada relevante para identificar as classes das questões.

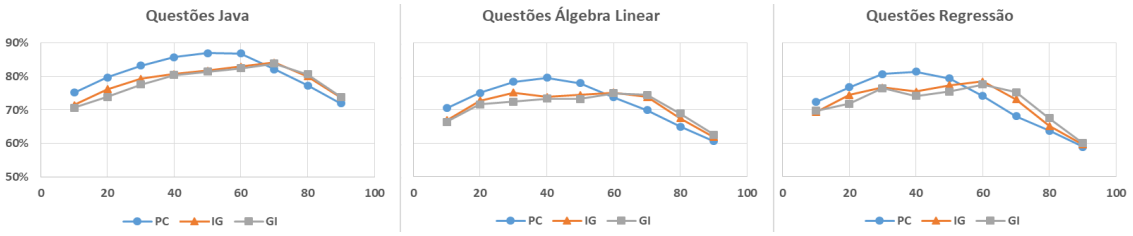


Figura 6. Acurácia X Filtro Naive Bayes

Outro ponto importante consiste em analisar como as características extraídas foram pesadas em cada grupo de questões e métodos de cálculo de peso. A Tabela 4 apresenta a posição da característica dentro do conjunto total. A primeira observação que pode ser feita é que a maioria das características permaneceu na metade superior do conjunto. Dentre as propriedades que estiveram sempre presentes nos subconjuntos com melhor acurácia estão: Tamanho da Descrição, Contagem de *Tags*, Erros de Linguagem, Legibilidade, Polidez, Número de Questões Postadas e Contagem de URL. Por outro lado, apenas a característica Seno Dia foi considerada ruído em todos os grupos de questões. Pode-se verificar também que não há uma equivalência nos pesos gerados pelos diferentes métodos. Por exemplo, nas questões de Álgebra Linear a característica Número de Respostas Dadas foi considerada ruído pelo peso Correlação de Pearson e, em contrapartida, foi considerada a 3ª mais importante tanto para *Information Gain* quanto para *GINI Index*.

A Tabela 4 também ajuda na validação da proposta em relação à filtragem das características. Por exemplo, a propriedade Há Código Na Descrição foi considerada relevante para as Questões Java e Álgebra Linear, porém para Regressão foi apontada como ruído. Outros exemplos seriam Número de Respostas Dadas e Proporção de Respostas/Perguntas consideradas relevantes apenas em Questões Java.

Aplicou-se o teste t para duas amostras pareadas com as seguintes hipóteses: H0: em média as abordagens WANQA e Baseline x tem mesma acuracidade e H1 (em média, a abordagem WANQA tem maior acurácia que o Baseline x). Obteve-se os seguintes valores p: (WANQA/B2)=0,0098 – (WANQA/B3)=0,0189 – (WANQA/B1)=0,0701. Considerando o nível de significância em 5%, pode-se rejeitar as hipóteses de igualdade em relação aos baselines 2 e 3. No baseline 1 não é possível a mesma afirmação, para re-

jeitar a hipótese teria que considerar o nível de significância em 10%. Ressalta-se que o baseline 1 foi proposto para trabalhar com questões do StackOverflow.

Tabela 4. Ranking dos Pesos das Características Extraídas

Característica	# Java (7670)			# Álg. Linear (2865)			# Regressão (3530)		
	PC	IG	GI	PC	IG	GI	PC	IG	GI
Tamanho da Descrição	1	4	3	3	4	4	5	1	1
Contagem de *tags*	2	14	13	1	1	1	6	10	9
Há Código na Descrição	3	11	11	72	347	244	2764	3530	3530
Erros de Linguagem	5	5	5	2	5	5	14	41	35
Título Inicia com Palavra WH	6	30	28	91	542	317	1676	3528	3528
Número de Questões Postadas	7	9	10	191	21	16	1341	593	453
Contagem de Palavras WH no Título	9	35	33	73	345	234	1703	1182	1304
Número de Respostas Postadas	14	2	2	1915	3	3	3278	7	7
Legibilidade	21	25	23	4	12	10	525	201	160
Proporção de Respostas/Perguntas	24	1	1	2142	2	2	2953	6	6
Polidez	36	110	95	71	235	165	10	8	8
Contagem de Palavras WH na Descrição	38	206	184	155	224	159	2306	2127	1213
Número de Dias Como Membro	54	6	7	23	13	11	3442	105	72
Tamanho do Título	203	308	268	2818	247	220	64	14	14
Contagem de URLs	434	1336	1091	32	122	99	640	583	404
Subjetividade	3737	999	784	265	7	6	1400	73	60
Seno Hora	16	101	87	1654	552	357	1382	1065	732
Cosseno Hora	207	363	308	136	159	105	122	328	243
Seno Dia	6588	7670	7670	2111	1861	1110	3143	3529	3529
Cosseno Dia	201	1227	880	177	601	476	1056	2272	2201

4. Trabalhos Relacionados

Nesta seção são apresentados e comparados trabalhos relacionados à tarefa de classificação de questões. Os trabalhos citados são abordagens que analisam o conteúdo das comunidades para identificar os motivos pelos quais parte das questões não possuem respostas. Além disso, as abordagens propõem identificar que as questões receberão ao menos uma resposta, que é também o objetivo desta proposta.

Asaduzzaman et al.[Asaduzzaman et al. 2013] estudaram os motivos de uma questão permanecer sem respostas na comunidade StackOverflow. Nesse trabalho, uma questão não respondida significa que não recebeu respostas até um mês depois da postagem. Os resultados obtidos não foram muito satisfatórios, mas as características analisadas serviram como ponto de partida para elaboração de abordagens de classificação de questões. Saha, Saha e Perry [Saha et al. 2013] analisaram toda a base de perguntas e respostas disponível no StackOverflow. A análise das questões foi feita empregando duas medidas: *Information Gain* e *Information Gain Ratio* sobre 12 características. A partir dos valores encontrados, efetuaram a classificação das questões com dois grupos de características. No primeiro grupo estavam somente as 6 características com as medidas mais altas e no segundo todas as características. A conclusão dos autores é que a diferença nos resultados dispensa o uso de todas as características disponíveis para identificar as questões que não estão respondidas na base.

Chua e Banerjee [Chua and Banerjee 2015] desenvolveram um *framework* para explicar porque algumas questões são respondíveis e outras não. Foram analisados os

metadados e conteúdo das questões. O conjunto de dados utilizado foi composto por 3000 questões sobre Java postadas no *site* StackOverflow. Algumas das inferências encontradas foram que as questões de usuários novos tendem a ser mais respondidas e que descrições breves ou poucas *tags* atraem respostas.

O trabalho de Yang et al. [Yang et al. 2011a] foi o dos primeiros a classificar questões em respondíveis ou não. A análise das questões foi feita usando tópicos latentes descobertos nos textos e características adicionais, como tamanho da questão e hora da postagem. Os autores declaram que os resultados ainda são ruins para uso prático. Baseando-se no trabalho de Yang et al. [Yang et al. 2011a], Dror, Maarek e Szpektor [Dror et al. 2013] buscaram algo ainda mais específico: predizer quantas respostas uma nova questão vai receber. A abordagem proposta consiste em avaliar características e vetores de palavras através de um conjunto de modelos de subárvores pois as categorias na comunidade estudada estão organizadas em estrutura de árvore. Dessa forma, a classificação é feita com a combinação de regressões logísticas dos nodos folha e respectivos nodos superiores. Porém, essa combinação de modelos só deve ser vantajosa quando as categorias estão organizadas em árvore. O estudo que mais se aproxima da proposta deste artigo é o de Fong, Zhou e Moutinho [Fong et al. 2015]. A proposta dos autores foi usar *feature selection* com aplicação de uma meta heurística chamada *Accelerated Particle Swarm Optimization* (APSO)[Yang et al. 2011b]. A técnica consiste em treinar os modelos iterando com vários subconjuntos de características até que se atinja um valor esperado ou um certo número de rodadas. Com a aplicação do APSO obteve-se melhor acurácia. Porém, essa proposta trabalha de forma não determinística, ou seja, nem sempre os melhores subconjuntos de características serão selecionados.

A maioria dos trabalhos estudados não considera o uso das palavras escritas na composição das perguntas. Yang et al. [Yang et al. 2011a] usaram os termos para extração de tópicos, porém, o conjunto extraído não trouxe resultados de uso prático. Dror, Maarek e Szpektor [Dror et al. 2013] usaram a presença ou não dos termos na classificação, mas sem especializar o treinamento em cada categoria. Apenas um dos trabalhos [Fong et al. 2015] empregou seleção de características, diferente das outras propostas que usaram todo o conjunto. A seleção de características faz com que se elimine o ruído das propriedades irrelevantes e assim melhora a acurácia da classificação. Um outro ponto a ser observado, que causa diminuição na acurácia, é classificar ao mesmo tempo, questões de todas as categorias, como visto em [Yang et al. 2011a, Dror et al. 2013, Asaduzzaman et al. 2013, Saha et al. 2013]. O treinamento de cada categoria criando um vetor *tf-idf* traz melhor acurácia, conforme visto nos experimentos, pois os termos mais pertinentes aos assuntos das categorias são ressaltados.

5. Conclusões e Trabalhos Futuros

Neste artigo foi apresentada uma proposta para a classificação de novas questões postadas em CQAs como respondíveis ou não respondíveis. Diferente de trabalhos anteriores estudados, buscou-se apresentar um método aplicável à maioria das CQAs. Os resultados apresentados pelos experimentos confirmam que a adição de um vetor *tf-idf* e a filtragem das características de acordo com os seus pesos melhoram a acurácia dos modelos gerados. Com os resultados obtidos nos experimentos, é possível definir Naive Bayes e SVM como bons classificadores para o problema. Para cálculo dos pesos, na definição da relevância das características, ficou evidente nos resultados apresentados que a Correlação

de Pearson foi melhor que as demais utilizadas. Como trabalhos futuros, pretende-se elaborar um modo automático para definir o tamanho de subconjunto que traz melhor acurácia e testar conjuntos de questões de categorias mais diversas às dos experimentos, como, por exemplo, política ou saúde.

Referências

[Aggarwal 2015] Aggarwal, C. C. (2015). Mining text data. In *Data Mining: The Textbook*, chapter 13, pages 288–291;429–433. Springer Publishing Company, Incorporated.

[Asaduzzaman et al. 2013] Asaduzzaman, M., Mashiyat, A. S., Roy, C. K., and Schneider, K. A. (2013). Answering questions about unanswered questions of stack overflow. In *Proceedings of the 10th Working Conference on Mining Software Repositories*, MSR '13, pages 97–100, Piscataway, NJ, USA. IEEE Press.

[Baeza-Yates and Ribeiro-Neto 2008] Baeza-Yates, R. and Ribeiro-Neto, B. (2008). *Modern Information Retrieval*. Addison-Wesley Publishing Company, USA, 2nd edition.

[Chua and Banerjee 2015] Chua, A. Y. and Banerjee, S. (2015). Answers or no answers: Studying question answerability in stack overflow. *Journal of Information Science*, 41(5):720–731.

[Dror et al. 2013] Dror, G., Maarek, Y., and Szpektor, I. (2013). Will my question be answered? predicting "question answerability"in community question-answering sites. In *Proceedings of the 2013th European Conference on Machine Learning and Knowledge Discovery in Databases - Volume Part III*, ECMLPKDD'13, pages 499–514, Berlin. Springer.

[Fong et al. 2015] Fong, S., Zhou, S., and Moutinho, L. (2015). Text analytics for predicting question acceptance rates. *IT Professional*, 17(4):34–41.

[Saha et al. 2013] Saha, R. K., Saha, A. K., and Perry, D. E. (2013). Toward understanding the causes of unanswered questions in software information sites: A case study of stack overflow. In *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering*, ESEC/FSE 2013, pages 663–666, New York, NY, USA. ACM.

[Srba and Bielikova 2016] Srba, I. and Bielikova, M. (2016). A comprehensive survey and classification of approaches for community question answering. *ACM Transactions on the Web*, 10(3):18:1–18:63.

[Yang et al. 2011a] Yang, L., Bao, S., Lin, Q., Wu, X., Han, D., Su, Z., and Yu, Y. (2011a). Analyzing and predicting not-answered questions in community-based question answering services. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, AAAI'11, pages 1273–1278. AAAI Press.

[Yang et al. 2011b] Yang, X.-S., Deb, S., and Fong, S. (2011b). Accelerated particle swarm optimization and support vector machine for business optimization and applications. *Networked Digital Technologies*, pages 53–66.

[Zhou and Fong 2016] Zhou, S. and Fong, S. (2016). Exploring the feature selection-based data analytics solutions for text mining online communities by investigating the influential factors: A case study of programming cqa in stack overflow. In *Big Data Applications and Use Cases*, pages 49–93. Springer.

SmartLTM: Smart Larger-Than-Memory Storage for Hybrid Database Systems

Paulo R. P. Amora[1], **Elvis M. Teixeira**[1],
Francisco D. B. S. Praciano[1], **Javam C. Machado**[1]

[1]Laboratório de Sistemas e Bancos de Dados (LSBD)
Computer Science Dept – UFC – CEP 60440-900 – Fortaleza – CE – Brazil

{paulo.amora,elvis.teixeira,daniel.praciano,javam.machado}@lsbd.ufc.br

Abstract. *Main-memory DBMS can offer hybrid and evolving storage architectures, instead of the traditional row or column storage layouts. Even if RAM is affordable nowadays, it is still a limited resource concerning available storage space in comparison to conventional storage devices. Due to this space restriction, techniques that leverage a trade-off between storage and query performance were developed and should be applied to data that is not frequently accessed or updated. This work proposes SmartLTM, a data eviction mechanism that considers the decisions previously taken by the DBMS in optimizing data storage according to query workload. We discuss how to migrate data, access it and the main differences between our approach and a row-based one. We also analyze the behavior of our solution in different storage media. Experiments show that cold data access with SmartLTM incurs an acceptable 17% of throughput loss, against 26% of the row-based one, while retrieving only half of the data to answer queries.*

1. Introduction

One of the great challenges in database management is how data should be available. As time passes by, newer data usually has more importance than older data, with a few exceptions, data becomes stale after a period of time. Current database systems setups usually work by having this new, more used data available in an OLTP database and the old, more stale data in an OLAP data warehouse, through the process of data migration and the Extract, Transform, Load mechanism. This setup brings several drawbacks, such as not having actual access to a real-time data analytics, because not all data is present in both places, and the maintenance cost of keeping two separate infrastructures may hinder some applications.

As technology improves, data processing requires more speed because organizations such as businesses or research labs acquire and accumulate data at a very rapid pace and this data must be processed into information promptly, which may be too fast for current database storage engines. This scenario creates a new kind of workload, called HTAP (Hybrid Transactional Analytic Processing), characterized by having both transactional and analytic features [Grund et al. 2010] [Kemper and Neumann 2011] [Alagiannis et al. 2014] [Appuswamy et al. 2017].

Traditional relational DBMS architectures do not handle well this kind of workload, because they adopt a fixed form of storing data contained in tables. Be it the row-based for OLTP databases, where tuples are stored contiguously, and is optimized by

design for workloads that access only a small range of tuples, but many attributes of each tuple; or column-based, where the attributes of several tuples are stored contiguously, and responds well to range and aggregation queries on single attributes. A proposed Flexible Storage Model (FSM)[Arulraj et al. 2016] aims to handle both workloads more optimally, having a mixture of both. The motivation for this FSM is to allow the retrieval of more relevant data, avoiding wasting cache space with data that won't be used in query processing. This model can be generated incrementally, using the query workload and accessed attributes as a clue on how to optimally organize and present data, making better use of upper layers of memory, such as CPU caches.

This data transformation is a costly task to perform on a slow, larger storage device, which is why the databases that execute this kind of transformation, be it a fixed one or an adaptive one, are usually main-memory databases. RAM is still a limited resource, and databases must be mindful not to overuse it, as data is also stored alongside structures like indexes and other auxiliary mechanisms.

Not all data are relevant to database users, in fact, usually, the most recent data is queried and modified, and as time passes by, those tuples become stale, except for a few attributes and only on aggregate queries. Note that this is not true for all entities in the system, for example, in a sales business setting, this behavior can be observed on orders, but not on stock warehouses, which may frequently be updated or items, that are mostly immutable, but often point queried for reads.

This skewed access pattern allows optimization concerning storage space. Data that is not being accessed, nor will be accessed, named cold data, can be moved out to larger, slower storage media, while preserving the working set, also called hot data, not to hinder transactional throughput performance. Data locality should also be transparent to the DBMS upper layers, to avoid specialized code and unnecessary overheads. Project Siberia[Eldawy et al. 2014] divides this problem into 4 categories, cold data classification, cold data storage, cold data access reduction and cold data access and migration.

This work focus on 3 of the 4 categories. It proposes a novel way to store cold data, applies techniques to avoid unnecessary cold data access and discuss how to access cold records. In summary, the main contributions of this work are:

- A cold data storage that considers the hybrid organization
- An application of techniques to avoid cold data access
- A performance study of the impact of storage media on our approach

We prototyped SmartLTM within PelotonDB[Pavlo et al. 2017], a main-memory hybrid database system and performed our evaluation using benchmarks from OLTPBench[Difallah et al. 2013]. With a reasonable amount of cold data access (50%), we find that the performance decrease is around 17% in a high-performance SSD. More experiments and results are presented further in the paper.

This paper is organized, as follows: Section 2 details SmartLTM and discuss design decisions. Section 3 explains how our approach is integrated within the DBMS. Section 4 presents the experimental evaluation. Section 5 briefly discusses related works, and we conclude in Section 6.

2. SmartLTM

2.1. Background

Instead of the traditional main-memory storage, we introduce a cold storage component inside the database architecture. The cold storage is separate from the main memory storage. Data is moved to the cold storage through a process called eviction, which can be defined as the inverse of caching. While caching keeps data frequently used in a faster medium, eviction moves data infrequently used to a slower medium. However, differently from other works that employ eviction, like Anti-caching[DeBrabant et al. 2013] and Siberia[Eldawy et al. 2014], the storage architecture is hybrid instead of row-based. One example of a hybrid storage DBMS is Peloton. In Peloton, data is organized according to the tile architecture[Arulraj et al. 2016], a specific in-memory organization to make data more available to the execution engine, and it has a few important definitions and terms that will be used throughout the article.

A table is composed of a list of tile groups, which can be seen as horizontal partitions within the table. A tile group has a fixed limit of tuples, the same schema as the table but it is composed of a disjoint set of tiles.

A tile is akin to a vertical and horizontal partition of a table, which contains a subset of the attributes as its schema, as well as only the subset of tuples of the table enclosed by the tile group. Different tile groups may have different tile layouts.

2.2. Data Eviction mechanism

Rather than doing data eviction one tuple at a time like Anti-caching, the mechanism uses a coarser granularity, evicting whole tile groups. For a tile group to be a candidate for eviction, it must not have been directly accessed, for read or write queries. Once the tile group is full, it will be accessed only for reads, however, if data that is being frequently accessed is evicted, the DBMS performance will decrease drastically.

The eviction process runs in a background thread and while the data is being written out, read transactions can still access it in memory if needed. Once data is written out, it is removed from main memory as soon as the older transactions cease to use it. It starts when a given threshold is reached.

Data is evicted in a format inspired by works like PAX[Ailamaki et al. 2002] and NoDB[Alagiannis et al. 2012]. It is written to the secondary storage in separate files, following the tile layout generated according to the workload. This approach respects the work previously done by the DBMS in organizing data in an optimal arrangement, allows for parallel retrieval of data in supporting devices, like SSDs, and preserves together data that is accessed together, reducing wasteful I/Os. To be able to skip unnecessary data, we also need to write out the column map, which maps the schema columns to the corresponding tile and offset.

Algorithm 1 details the execution of the eviction process.

2.3. Cold Storage

The cold storage is persistent storage where evicted data resides. It is decoupled from the database storage and non-transactional because it is a file. Tuples present within

Algorithm 1: Eviction Algorithm

```
Data: Tile groups in the table
Result: Tile groups evicted from the table and auxiliary structures
for tile group in table do
    if tile group is marked for eviction then
        write column map to external storage;
        create SMA for tile group;
        for tiles in tile group do
            add data to cuckoofilter;
            write tile to external storage;
        end
        delete tile group from table;
    end
end
```

expelled tile groups are read-only and not modifiable, although they can be invalidated in-memory, in case of deletion. Section 3 describes the behavior of operations with the new architecture.

2.4. Access Filters

Once data is moved to cold storage, it should be accessed only when needed, given that secondary storage operations are expensive in comparison to main memory. Some structures help to avoid unnecessary access to cold storage. Bloom Filters[Bloom 1970] is a non-deterministic structure that can tell if an element is absent or if it may be present, but here we employ Cuckoo Filters[Fan et al. 2014], an evolution of Bloom filters that is more space-efficient and supports delete operations.

Cuckoo Filters are hash-based. Therefore they only support equality comparisons. Another structure called Small Materialized Aggregates (SMA)[Moerkotte 1998], also known as Zone Maps, aids when checking for the presence of data in ranges, to avoid bringing to main memory all the data in the cold storage. It consists of precomputed aggregates (max, min) for tile groups (horizontal partitions), which can tell if a given key or range is present in that partition. There is an implementation of SMAs present in Peloton, for memory resident tile groups.

The CuckooFilters and SMAs are stored in main memory, alongside hot data. The storage space they occupy is negligible in comparison to the space saved.

2.5. Data Retrieval Mechanism

When a query is posed against the DBMS, it must try to answer the query with the memory resident data. For example, if the query asks for an exact match in a unique column. If the answer is not sufficient or not found, the cold storage must be probed, to check if there is a possibility that data present in the cold storage might answer the query. From the probe, two scenarios are possible.

If it's deemed that the cold storage cannot answer the query, then, the DBMS returns an answer to the client, saving an expensive cold access. On the other hand, the

probe returns the candidate tile groups in cold storage that may contain the data. Those candidate tile groups are then retrieved from the cold storage, but not entirely. Only the tiles containing the queried attributes are returned, as the column map links the queried columns and the correct tiles and offsets. After the retrieval, the data is validated against the Cuckoo Filter for deletes, and they are reassembled in a temporary in-memory table, that is disposed of when the transaction is completed. Algorithm 2 clarifies the data retrieval mechanism.

Algorithm 2: Data retrieval algorithm

```
   Data: Columns accessed, candidate tile groups
   Result: Temporary structure containing requested data
1  for tile group in candidate tile groups do
2  |  retrieve column map;
3  |  tiles = column map[columns accessed];
4  end
5  for tile in tiles do
6  |  retrieve columns accessed;
7  |  create temp table;
8  |  add retrieved tuples to temp table;
9  end
```

While data is being retrieved from the cold storage, the current transaction waits for the data. This retrieval does not bring many concurrency issues because of multi-version concurrency control.

2.6. Discussion

By dividing possibly large files into smaller ones, the I/Os become less costly and devices that allow efficient random access benefit from this. By keeping together data that is accessed together, the I/Os are not wasteful, avoiding the useless retrieval and load of data that is not necessary to the current query. The data retrieval mechanism ensures that cold access is done only when necessary. The synchronous retrieval is preferred according to [Ma et al. 2016].

3. Integration with the DBMS

Inserts. New tuples are always inserted in main memory. It is assumed that because they are new data, they will frequently be accessed and it is not for the benefit of performance to add new tuples directly in the cold storage.

Deletes. Deletes in main memory data happen as usual. When deletes happen in cold storage data, the respective entry is removed from the filter, but no access is made to the cold storage. This removal effectively makes the record inaccessible in cold data while avoiding access.

Updates. Updates with relation to cold data are nothing more than a delete operation followed by an insert. Meaning that the updated record will always be in main memory. This new version may be evicted later to cold data. Updated data is considered hot because new data is always considered hot.

Reads. Reads can be seen as two types of queries: point queries, which are reads done through an equality predicate and range queries, which use a more flexible predicate, like *less than* or *greater than*. A broader view of reads has already been presented in section 2.5. Reads are also benefited from new data being only placed in main memory. When a transaction with a read validates, it checks for changed or new data, which is already present in main memory, avoiding cold storage accesses.

Point queries. Point queries are first posed against the hot data. If the predicate is a primary key or unique, the query may be answered only by looking at data present in main memory. If it is not completely answered, the predicate is evaluated by the CuckooFilter, which can tell if the data is not present in the cold storage. In the end, if the probe determines that data may be present in the cold storage, we move to data retrieval.

Range queries. Range queries are first posed against the hot data. If it is not completely answered, data may be present in the cold storage, which is probed using the SMAs, since they are suitable for ranges, we move to data retrieval.

4. Experimental evaluation

To evaluate our strategy, we prototyped it in Peloton, a hybrid, main-memory, multi-versioned DBMS. Besides adding the new components, a few changes were made in the engine to integrate the new components with the query processing engine. The logging and garbage collection components were disabled to avoid interference with other secondary storage media and ensure tile group immutability.

4.1. Setup

The experiments were executed in an Intel Core I7 7800X with 6 physical cores and hyperthreading, with 64GB of RAM and 8MB of L3 cache with Ubuntu 16.04 LTS as the OS. Two different storage devices were selected as cold storage, a commodity WD Blue SATA 3 7200rpm HDD and an Intel DC P3600 SSD. The HDD has a reported IOPS of 500 and the SSD 230000 for random read operations. To ensure maximum parallelism and avoid interference from context switch, the number of threads executing queries against the database is purposely low.

4.2. Benchmarks

The benchmark selected is YCSB[Cooper et al. 2010], due to the easiness of keeping track of operations. We used OLTPBench to execute the benchmark but modified some operations. While we kept the semantics of the key uniqueness, the primary key constraint was disabled, and no indexes whatsoever were created. The queries were also modified to diversify attribute access. Five queries selecting some columns were placed and randomly selected alongside the values, to allow a better data layout organization and effectively test the different organizations. They are shown below, with the layout organization after execution:

- Q_1: `SELECT f0 WHERE KEY =` X`;`
- Q_1: `[KEY][f0][f1, f2, f3, f4, f5, f6, f7, f8, f9]`
- Q_2: `SELECT f2, f4 WHERE KEY =` X`;`
- Q_2: `[KEY][f0, f1][f2, f3, f4][f5, f6, f7, f8, f9]`
- Q_3: `SELECT f1, f2, f3 WHERE KEY =` X`;`
- Q_3: `[KEY][f0][f1, f2, f3][f4, f5, f6, f7, f8, f9]`

- Q_4: `SELECT f1, f2, f6, f7 WHERE KEY =` X`;`
- Q_4: `[KEY][f0][f1, f2][f3, f4, f5][f6, f7][f8, f9]`
- Q_5: `SELECT f0, f1, f5, f8, f9 WHERE KEY =` X`;`
- Q_5: `[KEY][f0, f1][f2, f3, f4][f5][f6, f7][f8, f9]`

To have more control of cold and hot data accesses, we also added a uniform distribution to select key values, alongside the standard Zipfian. With a uniform distribution, cold storage access can be correctly estimated and is directly proportional to the amount of data evicted to the cold storage.

4.3. Workloads

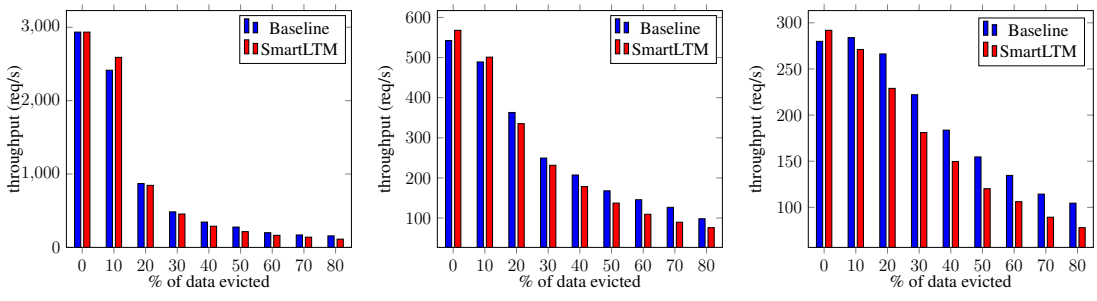
We defined four workloads to be executed, a read-only, an insert-only, a delete-only and an update-only. The scenario of data eviction to a secondary, slower storage medium shifts the burden to read queries, given that all modifying operations, like delete, insert and update, happen only in-memory, according to the mechanism proposed. Therefore, mixed workloads would only alleviate the bottleneck imposed by cold storage reads. Due to the nature of YCSB, all queried values are within the range of data loaded in the table, so there are no missed queries, and every one of them returns an answer, except in the delete workload.

4.4. Experiments and Results

SmartLTM and the baseline are implemented inside Peloton. The main difference between SmartLTM and the baseline is how data is organized and how it is evicted. The baseline evicts the tile group completely, as a row-based layout, while our approach separates the tiles, as described above. Both implementations take advantage of the access filters implemented to avoid cold storage access, to ensure fairness. An effort was also made to ensure direct access to the storage media, trying to avoid buffered reads as much as possible.

4.4.1. Read-only Queries

This experiment evaluates the impact of our approach with a read-only workload. Three different scale factors in YCSB were used: 50, 250 and 500. The access pattern is uniform, to allow accuracy when estimating hot data accesses and cold data accesses.



(a) 50k Tuples **(b) 250k Tuples** **(c) 500k Tuples**

Figure 1. HDD results (higher is better)

The HDD results are shown in figure 1. It is observable that the performance decrease is exponential, and that the baseline performs a little better than our proposed approach. The exponential decrease in performance is due to the access speeds of the media (HDD), which becomes the bottleneck in performance.

The baseline performs better due to the nature of an HDD device. When reading a single file which was written in neighboring sectors, the arm only needs a single spin to read all the data. In our proposed approach, the tiles are written separately, keeping the data inside them contiguous, but having no control over how different tiles of the same tile group are recorded. If different tiles are required to answer a query, the device would have to do multiple scans, and it is known that random access in an HDD is costly.

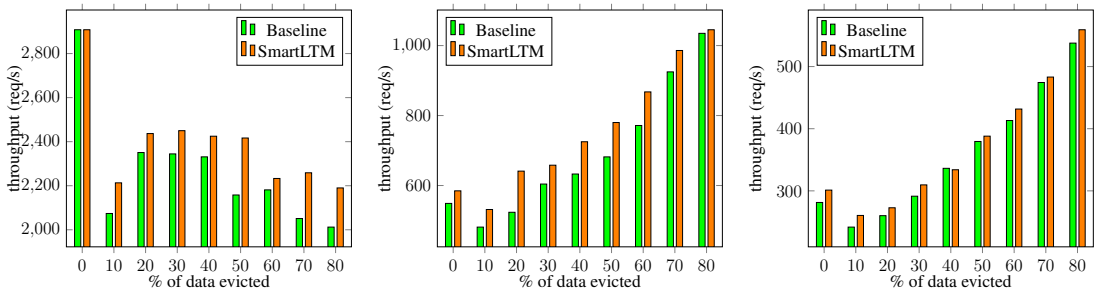


Figure 2. SSD results (higher is better)

The SSD results are shown in figure 2. The higher access speeds and random access behavior of the device demonstrates how our proposed approach is an improvement over the baseline, having a throughput loss of 17% vs. 26% of the baseline on figure 2a, at the 50% evicted data mark. It is also noted that the throughput increases as more data is evicted when there is a higher volume of data loaded into the database. From section 2.5, the in-memory sequential scan has a *O(n)* complexity and always happen. As less data is present in memory, the faster this scan happens. Given the SSD device's high speeds and that the cold storage read is also a targeted one, retrieving only the tiles where the queried attributes reside, the most expensive task becomes traversing all the in-memory data to search the value. Querying the cold storage is still an expensive operation, as observed in the 50k graph, where the cold storage read is visibly hindering the performance. It is also observable that the throughput increase follows an approximately linear pattern, more clearly seen in the 500k graph.

The random access optimization of SSDs makes clear that our approach is better than the baseline, especially where it most counts when the bottleneck becomes the cold data access. It can be verified that the throughput with our proposed approach almost doubles the one in the baseline, as more data is evicted, shown in figure 2a.

4.4.2. Insert, Update and Delete queries

This experiment evaluates the impact of SmartLTM with insert, update and delete workloads. Since those operations do not care about the cold storage, only probing it through the access filters, only one scale factor in YCSB was used, 250. The access pattern is

uniform, to allow accuracy when estimating hot data accesses and cold data probes.

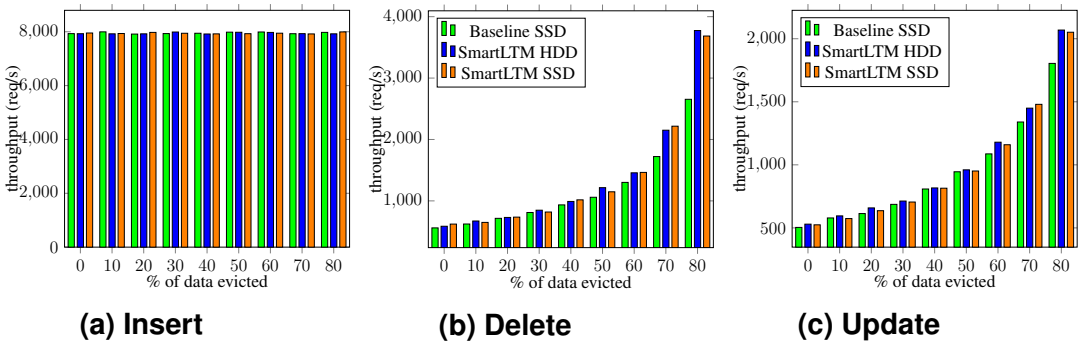


Figure 3. Insert, delete and update results

Figure 3 shows three separate results, one from an insert-only workload, that is observed to be almost constant, independent of how much data was evicted from the database. The proximity of insert results happens because inserts do not care about data that is present or absent in main memory, it only allocates a tuple and inserts the values. The update-only workload shows a throughput increase as data is evicted, clearly because, before updating, a sequential scan must happen to in-memory data. The probe in cold storage is a cheap operation because only the access filters are queried, and all the modifications are done in-memory. The delete-only workload behaves like the update-only, however, deleting a record involves fewer operations than updating, which is why the overall throughput is higher. It also can be observed that the baseline suffers on sequential scans. This behavior is an effect of the tile layout organization, which doesn't happen in the baseline.

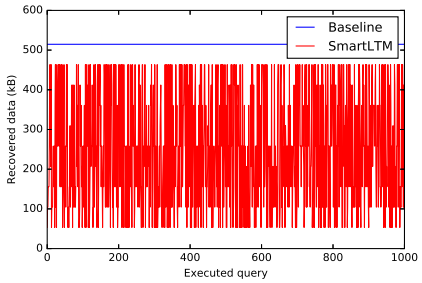
4.4.3. Retrieved data from disk

This experiment evaluates the impact of SmartLTM for retrieved data from disk. The read-only workload is executed with YCSB scale factor 250. The access pattern is uniform.

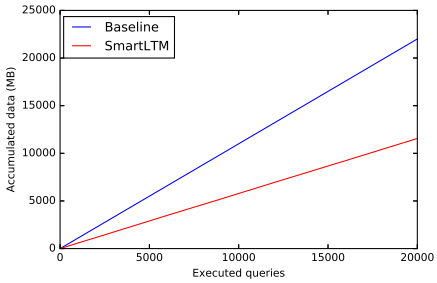
Figure 4 shows three separate results. Figures 4a and 4b are measurements taken with 50% of data evicted, and show how much data is retrieved per executed query and the evolution of how much data is retrieved during the benchmark execution. Figure 4c summarizes how much data was retrieved from disk in each eviction percentage and compares it to the baseline. Baseline retrieves a fixed amount because the tile group is evicted as a whole, while SmartLTM takes into account the previous organization, as described before. This experiment shows clearly that our approach is effective regarding data retrieval, by avoiding useless data to answer queries, based on the data organization.

5. Related Works

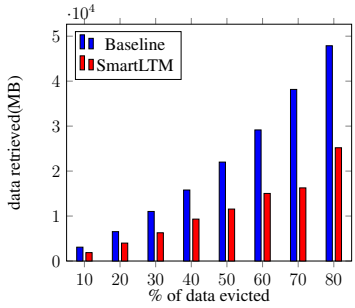
There are a few different solutions to in-memory space saving. Garbage collection (GC) is a well-known approach, adopted in several multi-version DBMS. Wu et al. [Wu et al. 2017] conduct a study of various techniques. GC by itself can save space by reusing invalid tuple slots. However, this brings an unwanted side effect of mixing hot and cold data. If a cold data partition receives one hot record, it may not be considered completely cold anymore.



(a) Data retrieved per query



(b) Accumulated data



(c) Total data retrieved

Figure 4. Data Retrieval results (lower is better)

To keep away from this side effect, the GC may not recycle used tuple slots, but be associated with another technique, called compaction, which can place data together and close any empty spaces. The challenge imposed by compaction in a hybrid storage DBMS environment is that data in different partitions may be organized in different ways, and mixing them would implicate a decision that results in some layouts being suppressed in favor of the many.

Aside from GC and compaction, compression is seen as the next step in space saving. Compressed data occupies a fraction of the original storage space, and while it introduces overhead to decompress data, everything is still present in main memory. Works like HyPer [Kemper and Neumann 2011] present an HTAP database that makes use of compression techniques to store unaccessed data better. An evolution of this storage model, called Data Blocks[Lang et al. 2016] optimizes even further the access speed to compressed data and the compression rates. Compression is the last stand when it comes to main memory storage.

However, main memory storage is finite. When data does not physically fit in main memory, secondary storage media is needed. Works like Anti-caching[DeBrabant et al. 2013] and Project Siberia[Eldawy et al. 2014] provide two different solutions.

Anti-caching tracks the tuple eviction candidates through an LRU chain, and when eviction is needed, the last members of the chain are written in a block back to disk. Evicted tuples are tracked through a specific table, containing the block id and tuple offset to evicted tuples. To access evicted data, a special pointer called a tombstone is placed when a given tuple is evicted. Transactions are processed making use of a pre-pass phase, which checks if any tombstone is accessed. If anything evicted may be accessed, the transaction aborts and a subroutine that brings the evicted tuples back to memory starts.

Then, the transaction is restarted and executes as usual. Since H-Store executes only one thread per execution node, this frees the thread to run other transactions while data is being retrieved.

Siberia tracks the eviction candidates offline by logging record accesses and sampling those logs to extract estimates of eviction candidates. Those are migrated to the cold storage when a user-defined threshold is achieved. Data retrieval from the cold storage is synchronous, and to access evicted data Bloom Filters are used.

LSM Trees [O'Neil et al. 1996] are also a data structure appropriate to larger-than-memory scenarios, but is more applied in key-value DBMS, and may not perform well in relational databases as the main storage mechanism.

Siberia and Anti-caching focus on an OLTP environment, and evict the data as tuples, while SmartLTM focuses on an HTAP environment and preserve the optimization that this kind of DBMS provides.

6. Conclusions and future work

In this work, we propose SmartLTM, a new, smarter way of executing data eviction concerning HTAP databases. Current data eviction solutions are designed for row-based DBMS and perform sub-optimally when adapted as-is to the new architecture, provided good random access secondary storage. Our experiments show that SmartLTM does not affect insert, delete and update response times, and achieve better response times while retrieving fewer data from secondary storage.

As future work, a global cache can be implemented containing the most accessed tile groups. Keeping the cold storage in modern, byte-addressable non-volatile memories (NVRAM) can also drastically improve performance since the retrieval would not need to bring useless data contained in the current storage, which is block-addressable.

Acknowledgements

This research was partially supported by FUNCAP/CE-Brazil (Grant BMD-0008-01237.01.09/17) and LSBD/UFC. I'd also like to thank Prof. Andy Pavlo and the CMU Database Group for his feedback and support during the early conceptual stages of this work.

References

Ailamaki, A., DeWitt, D. J., and Hill, M. D. (2002). Data page layouts for relational databases on deep memory hierarchies. *VLDB J.*, 11(3):198–215.

Alagiannis, I., Borovica, R., Branco, M., Idreos, S., and Ailamaki, A. (2012). Nodb: efficient query execution on raw data files - read. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2012, Scottsdale, AZ, USA, May 20-24, 2012*, pages 241–252.

Alagiannis, I., Idreos, S., and Ailamaki, A. (2014). H2O: a hands-free adaptive store - read. In *International Conference on Management of Data, SIGMOD 2014, Snowbird, UT, USA, June 22-27, 2014*, pages 1103–1114.

Appuswamy, R., Karpathiotakis, M., Porobic, D., and Ailamaki, A. (2017). The case for heterogeneous HTAP. In *CIDR*. www.cidrdb.org.

Arulraj, J., Pavlo, A., and Menon, P. (2016). Bridging the archipelago between row-stores and column-stores for hybrid workloads. In *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pages 583–598.

Bloom, B. H. (1970). Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13(7):422–426.

Cooper, B. F., Silberstein, A., Tam, E., Ramakrishnan, R., and Sears, R. (2010). Benchmarking cloud serving systems with YCSB. In *SoCC*, pages 143–154. ACM.

DeBrabant, J., Pavlo, A., Tu, S., Stonebraker, M., and Zdonik, S. B. (2013). Anti-caching: A new approach to database management system architecture. *PVLDB*, 6(14):1942–1953.

Difallah, D. E., Pavlo, A., Curino, C., and Cudré-Mauroux, P. (2013). Oltp-bench: An extensible testbed for benchmarking relational databases. *PVLDB*, 7(4):277–288.

Eldawy, A., Levandoski, J. J., and Larson, P. (2014). Trekking through siberia: Managing cold data in a memory-optimized database. *PVLDB*, 7(11):931–942.

Fan, B., Andersen, D. G., Kaminsky, M., and Mitzenmacher, M. (2014). Cuckoo filter: Practically better than bloom. In *CoNEXT*, pages 75–88. ACM.

Grund, M., Krüger, J., Plattner, H., Zeier, A., Cudré-Mauroux, P., and Madden, S. (2010). HYRISE - A main memory hybrid storage engine. *PVLDB*, 4(2):105–116.

Kemper, A. and Neumann, T. (2011). Hyper: A hybrid oltp&olap main memory database system based on virtual memory snapshots. In *Proceedings of the 27th International Conference on Data Engineering, ICDE 2011, April 11-16, 2011, Hannover, Germany*, pages 195–206.

Lang, H., Mühlbauer, T., Funke, F., Boncz, P. A., Neumann, T., and Kemper, A. (2016). Data blocks: Hybrid OLTP and OLAP on compressed storage using both vectorization and compilation. In *SIGMOD Conference*, pages 311–326. ACM.

Ma, L., Arulraj, J., Zhao, S., Pavlo, A., Dulloor, S. R., Giardino, M. J., Parkhurst, J., Gardner, J. L., Doshi, K., and Zdonik, S. B. (2016). Larger-than-memory data management on modern storage hardware for in-memory OLTP database systems. In *DaMoN*, pages 9:1–9:7. ACM.

Moerkotte, G. (1998). Small materialized aggregates: A light weight index structure for data warehousing. In *VLDB*, pages 476–487. Morgan Kaufmann.

O'Neil, P. E., Cheng, E., Gawlick, D., and O'Neil, E. J. (1996). The log-structured merge-tree (lsm-tree). *Acta Inf.*, 33(4):351–385.

Pavlo, A., Angulo, G., Arulraj, J., Lin, H., Lin, J., Ma, L., Menon, P., Mowry, T. C., Perron, M., Quah, I., Santurkar, S., Tomasic, A., Toor, S., Aken, D. V., Wang, Z., Wu, Y., Xian, R., and Zhang, T. (2017). Self-driving database management systems. In *CIDR*. www.cidrdb.org.

Wu, Y., Arulraj, J., Lin, J., Xian, R., and Pavlo, A. (2017). An empirical evaluation of in-memory multi-version concurrency control. *PVLDB*, 10(7):781–792.

Armazenamento Otimizado de dados RDF em um SGBD Relacional

Rafael L. Prado[1], **Rebeca Schroeder**[2], **Carmem S. Hara**[1]

[1]Departamento de Informática – Universidade Federal do Paraná - UFPR
CEP: 81530-900 – Curitiba – PR – Brazil

[2]DCC – Universidade do Estado de Santa Catarina - UDESC
CEP: 89219-710 – Joinville – SC – Brazil

rlprado@inf.ufpr.br, rebeca.schroeder@udesc.br, carmem@inf.ufpr.br

Abstract. *Several methods employ Relational Database Management Systems (RDBMS) to store RDF data. However, the direct mapping from RDF to a table of triples does not produce good query performance. This paper introduces AORR, an optimized method to store RDF data in a RDBMS. AORR identifies data entities in order to define a relational schema. In addition, AORR differs from related work by supporting SPARQL to SQL query translation as well as dynamic data insertions. An experimental study shows that AORR improves the overall query performance, compared to a close related work.*

Resumo. *Diversas propostas utilizam Sistemas Gerenciadores de Bancos de Dados Relacionais (SGBDRs) para o armazenamento de dados RDF. O mapeamento direto de RDF para uma tabela de triplas resulta em um desempenho ineficiente no processamento de consultas. Este artigo propõe AORR (**A**rmazenamento **O**timizado de dados **R**DF em SGBD**R**), um método que identifica entidades de dados para gerar tabelas. Além disto, AORR se diferencia de trabalhos relacionados por possibilitar a tradução de consultas SPARQL-SQL, bem como atualizações incrementais da base. Um estudo experimental mostrou que AORR apresenta desempenho superior em consultas, comparado a uma proposta alternativa que também adota o conceito de tabelas de entidades.*

1. Introdução

A Web Semântica surgiu do desejo de tornar a máquina capaz de interpretar as informações da Web, exigindo cada vez menos interação humana. Para esse fim, é necessário que haja uma padronização de como essas informações são acessadas e de como entidades do mundo real são identificadas. Nesse contexto, o W3C adotou o RDF (*Resource Description Framework*) como o modelo de dados padrão da Web Semântica e o SPARQL como sua linguagem de consulta. O RDF representa os dados como um conjunto de triplas (sujeito, predicado, objeto) sem que haja obrigatoriamente um esquema pré-definido. Sendo assim, dados RDF podem ser representados e armazenados em diferentes formatos. Embora existam diversas propostas para o armazenamento de RDF em sua forma nativa de grafo [Zeng et al. 2013, Penteado et al. 2015], o uso de um Sistema Gerenciador de Banco de Dados Relacional (SGBDR) não deve ser descartado para conjuntos de dados de até milhões de triplas [Zeng et al. 2013]. Uma das vantagens de utilizar um SGBDR é tirar proveito de todo o investimento em pesquisa e desenvolvimento feito na

sujeito	predicado	objeto
s1	p1	'joao'
s1	p2	Pessoa
s1	p3	'0000-0001'
s1	p3	'0000-0002'
s1	p4	s3
s2	p1	'fabiana'
s2	p2	Pessoa
s2	p3	'0000-0003'
s2	p3	'0000-0004'

sujeito	predicado	objeto
s2	p4	s4
s3	p5	'moto1'
s3	p2	Automotor
s4	p5	'moto2'
s4	p2	Automotor
s5	p5	'carro1'
s5	p2	Automotor
s5	p7	'32784738'
s6	p8	'Contato de Debora'

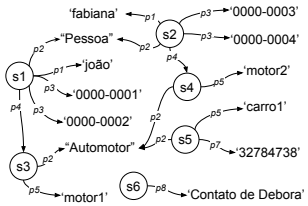


Figura 1. Base RDF em uma tabela SPO e sua representação em grafo

TB_FullPredicate

cs_identifier	tableName	overflowTable	columnName	predicate
CS1	PessoaRDF	<NULL>	nome	p1
CS1	PessoaRDF	<NULL>	type	p2
CS1	telefoneMultivalueRDF	<NULL>	telefone	p3
CS1	PessoaRDF	<NULL>	fk_veiculo	p4
CS2	AutomotorRDF	<NULL>	modelo	p5
CS2	AutomotorRDF	<NULL>	type	p2
CS2	<NULL>	Overflow_AutomotorRDF	cod_modelo	p7
Csover	<NULL>	Overflow	observacao	p8

TB_DatabaseSchema

cs_identifier	propertyName	valueType	tableName	tableAttribute
CS1	OID	literal	PessoaRDF	OID
CS1	nome	literal	PessoaRDF	nome
CS1	type	literal	PessoaRDF	type
CS1	veiculo	CS2	PessoaRDF	fk_veiculo
CS1	telefone	literal	telefoneMultivalueRDF	telefone
CS2	OID	literal	AutomotorRDF	OID
CS2	modelo	literal	AutomotorRDF	modelo
CS2	type	literal	AutomotorRDF	type
CS2	cod_modelo	literal	Overflow_AutomotorRDF	cod_modelo
Csover	observacao	literal	Overflow	pred

TB_Subj_OID

subj	OID	tableName
s1	1	PessoaRDF
s2	2	PessoaRDF
s3	3	AutomotorRDF
s4	4	AutomotorRDF
s5	5	AutomotorRDF
s6	6	Overflow

PessoaRDF

OID	nome	type	fk_veiculo
1	joao	Pessoa	3
2	fabiana	Pessoa	4

telefoneMultivalueRDF

OID	telefone
1	0000-0001
1	0000-0002
2	0000-0003
2	0000-0004

AutomotorRDF

OID	modelo	type
3	moto1	Automotor
4	moto2	Automotor
5	carro1	Automotor

Overflow

subj	pred	obj
6	'observacao	'Contato de Debora'

Overflow_PessoaRDF

subj	pred	obj

Overflow_AutomotorRDF

subj	pred	obj
5	'cod_modelo'	'32784738'

Figura 2. Exemplo de base relacional gerada pelo AORR

consolidação de sua tecnologia. A maneira mais direta de armazenar dados RDF em um SGBDR é no formato de triplas em uma tabela SPO, como ilustrado na Figura 1. O problema dessa abordagem é que o processamento de consultas SPARQL requer a execução de auto-junções sobre essa tabela, que tem cardinalidade igual a quantidade de triplas.

Exemplo 1: Considere a tabela T_{SPO} da Figura 1, onde são utilizados s_i e p_j para representar IRIs de sujeitos e predicados, respectivamente. Para facilitar a compreensão, uma representação equivalente em grafo é apresentada ao lado. Por exemplo, p_1 representa a IRI `http://xmlns.com/foaf/0.1/name` e p_2 representa a IRI `http://www.w3.org/1999/02/22-rdf-syntax-ns#type`. Uma consulta para obter os valores dos predicados p_1 e p_2 associados a um mesmo sujeito exigiria a execução de uma auto-junção de T_{SPO} sobre a coluna sujeito. Esta consulta pode ser expressa da seguinte forma em SPARQL: *select ?n ?t where {?s* $p1$ *?n . ?s* $p2$ *?t .}*, que resultaria em: *{('joao', Pessoa), ('fabiana', Pessoa)}*. □

Para minimizar a quantidade de auto-junções para o processamento de consultas e diminuir o volume de dados envolvidos nas operações, este artigo propõe o AORR (Armazenamento Otimizado de dados RDF em um SGBD Relacional). A estratégia adotada pelo AORR é criar tabelas de maior grau, contendo predicados frequentemente associados a um mesmo sujeito. Para permitir que consultas SPARQL possam ser traduzidas para consultas SQL sobre a base relacional, o AORR mantém informações sobre o mapeamento em tabelas de metadados.

Exemplo 2: A base relacional gerada pelo AORR para o armazenamento das triplas na Figura 1 está ilustrada na Figura 2. Esta base contém 4 categorias de tabelas: tabelas de metadados (*Tb_FullPredicate*, *Tb_DatabaseSchema*, *Tb_Sbj_OID*), tabelas de entidades (*PessoaRDF, AutomotorRDF*), tabelas de propriedades multivaloradas (*telefoneMultivalueRDF*) e tabelas de overflow. As tabelas de overflow são de dois tipos: overflow geral (*Overflow*) e overflow específico, uma para cada tabela de entidades (*Overflow_PessoaRDF, Overflow_AutomotorRDF*). O overflow geral é responsável por manter dados que não puderam ser identificados como entidades ou que gerariam tabelas muito pequenas. Em virtude da natureza semiestruturada do RDF, a tabela de overflow geral poderá conter muitos dados. Assim, as tabelas de overflow específico dividem esta carga ao manter atributos infrequentes que estão associados a uma entidade. As tabelas de metadados permitem que uma consulta SPARQL seja traduzida para SQL. Considere a consulta SPARQL do Exemplo 1. A tabela *Tb_FullPredicate* permite associar as IRIs dos predicados $p1$ e $p2$ a uma única tabela *PessoaRDF* e às colunas *nome* e *type*, respectivamente. Assim, a consulta pode ser processada apenas com uma projeção sobre *PessoaRDF*. □

Além de dar suporte à tradução de consultas, as tabelas de metadados, juntamente com as tabelas de overflow, permitem que o AORR seja capaz de realizar atualizações incrementais na base. O processo de geração da base relacional passa por três grandes etapas: extração de esquema, carga de dados e atualização da base. A etapa de extração de esquema da base relacional é baseada no processo ERSR [Pham et al. 2015]. Tanto o AORR como o ERSR são otimizados para processar consultas de busca por padrões básicos em grafos (consultas BGP, em particular consultas no formato estrela e flocos de neve [Aluç et al. 2014])[1]. Contudo, os dois processos possuem algumas diferenças. O ERSR não considera a existência de tabelas de overflow específicas, mas apenas um overflow geral. Além disso, o ERSR não dá suporte a atualizações incrementais e não propõe o armazenamento de metadados, que dá suporte à tradução de consultas. Nos resultados relatados pelo ERSR, o armazenamento de dados em tabelas de entidades apresentou desempenho superior no processamento de consultas de até 5 vezes, comparado a tabelas SPO [Pham et al. 2015]. Assim, neste artigo foi comparado o desempenho de processamento de consultas entre o AORR e o ERSR. O AORR obteve melhor desempenho em todas as consultas consideradas, sendo o maior ganho de 198%, enquanto no pior caso ainda houve ganho de 1,34%.

As contribuições desse trabalho são: uma proposta de mapeamento de dados RDF para relacional; suporte à atualização incremental da base RDF; armazenamento de metadados que possibilitam a tradução de consultas SPARQL para SQL; e uma análise experimental que mostra a eficiência do sistema proposto. O restante do artigo está organizado da seguinte forma. A Seção 2 discute outras abordagens que utilizam um SGBDR para o armazenamento de RDF. O AORR é apresentado na Seção 3. A análise experimental é descrita na Seção 4 e a Seção 5 finaliza o artigo apresentando trabalhos futuros.

2. Trabalhos Relacionados

Existem diversas propostas para utilizar um SGBDR para o armazenamento RDF. Elas podem ser divididas entre abordagens verticais e horizontais. Nas abordagens verticais cada linha da tabela armazena uma única tripla da base. Já na abordagem horizontal, cada linha

[1]Maiores detalhes sobre as consultas suportadas pelo AORR podem ser obtidos em [Pauluk et al. 2018]

contém um conjunto de triplas. Um exemplo de abordagem vertical são as tabelas SPO, da Figura 1. Dentre os sistemas que adotam esta forma de armazenamento estão o Virtuoso[2] e Jena[3]. Já a proposta de [Abadi et al. 2007] propõe a criação de uma tabela para cada predicado distinto encontrado na base, o que pode resultar em uma grande quantidade de tabelas. Os trabalhos de [Bornea et al. 2013], [Scabora et al. 2017], [Aluç et al. 2014], [Pham et al. 2015] e [Ramunajam et al. 2009] adotam a abordagem horizontal. Os dois primeiros optam por manter um conjunto de k pares (predicado, objeto) ligados ao mesmo sujeito em uma mesma linha da tabela, o que pode resultar em uma grande quantidade de valores nulos. O terceiro utiliza informações da carga de consultas para gerar o esquema. Já os dois últimos fazem a extração de um esquema relacional a partir da base RDF.

A proposta de [Ramunajam et al. 2009] para a extração do esquema relacional tem a preocupação de criar tabelas que possam comportar todas as triplas RDF, baseando-se principalmente nas classes associadas aos sujeitos. Já a proposta ERSR [Pham et al. 2015] cria um esquema baseado em vários critérios, como quantidade mínima de registros na tabela, quantidade máxima de tabelas e frequência mínima da predicados. O esquema do ERSR é obtido a partir da identificação de *Characteristic Sets* (CSs), que correspondem aos conjuntos de predicados que podem estar associados a sujeitos. Avaliações experimentais [MahmoudiNasab and Sakr 2010] mostram que a abordagem horizontal com a geração de um esquema relacional tradicional apresenta melhor desempenho sobre as demais alternativas. A abordagem de [He et al. 2017] corrobora com estes resultados ao avaliar o processamento de consultas SPARQL sobre um SGBDR formado a partir de um esquema RDF pré-definido.

O AORR tomou como base o processo de extração de esquema do ERSR, porém estende este processo com a introdução de tabelas de overflow específico associadas às tabelas de entidades. As tabelas de overflow específico atuam na redução da quantidade de tuplas do overflow geral, além de dar suporte à atualização incremental da base. No ERSR a geração da base relacional ocorre uma única vez, sendo necessário recriar a base relacional para comportar novos dados. Além disso, a estrutura de metadados mantida pelo AORR oferece suporte à transformação de consultas SPARQL para SQL.

3. Armazenamento Otimizado de dados RDF em um SGBDR

Esta seção apresenta o AORR, um método de **A**rmazenamento **O**timizado de dados **R**DF em um SGBD **R**elacional. A estratégia adotada pelo AORR consiste em extrair um esquema relacional no qual predicados frequentemente encontrados em conjunto dão origem a tabelas *de entidades*. A Seção 3.1 apresenta os processos de extração de esquema e carga da base relacional. Para dar suporte à natureza semi-estruturada das bases RDF, o AORR cria tabelas de overflow para comportar dados que não se adequam às tabelas de entidades. As tabelas de overflow adotam o esquema SPO e são de dois tipos: específico e geral, conforme detalhado na Seção 3.2. As informações de mapeamento da base RDF para relacional são mantidas em tabelas de metadados, como detalhado na Seção 3.3.

3.1. Geração da Base Relacional

O processo de geração da base relacional consiste de 7 passos, que compõem o Algoritmo 1. O algoritmo recebe como entrada uma base RDF B e um conjunto de parâmetros de

[2]https://virtuoso.openlinksw.com/

[3]https://jena.apache.org/

configuração, e gera como saída uma base relacional R. Tais parâmetros afetam diretamente a geração da base relacional, dado que o min_t e Ub_{tbl}, quantidade mínima de registros e quantidade máxima de tabelas, respectivamente, determinam se uma tabela será inserida no overflow geral (OverflowG). Já o T_{inf}, frequência mínima, configura o quão frequente uma propriedade deve ser, determinando assim se ela será mantida na tabela de entidade ou no overflow específico (OverflowE). Uma base RDF B é formada por um conjunto de triplas (s, p, o). Sobre B, é definida a função *pred*, que associa um sujeito ao seu conjunto de predicados. Ou seja, *pred(s)* = $\{\ p \mid (s, p, o) \in B\}$. Por exemplo, o conjunto de triplas da Figura 1 compõe uma base RDF, na qual *pred*(s_5) = *{p2, p5, p7}*.

Algoritmo 1: Geração da base relacional

Entrada: base RDF B, parâmetros de configuração T_{inf}, min_t, Ub_{tbl}
Saída: uma base relacional R

```
1 início
2     (S, M) := criaBaseCategorizada( B );
3     label := atribuiRotulo(S, M, T_inf);
4     juntaCS( S, M, label, T_inf );
5     map := geraMapeamento( S, M, label, B );
6     filtragem( S, M, map, T_inf, min_t, Ub_tbl);
7     R := criaTabelas( map );
8     carregaDados( R, map, B, S, M );
9 fim
```

Passo 1: Criação da base categorizada

A função *criaBaseCategorizada* (Linha 2) identifica os *Characteristic Sets* (CSs) da base, ou seja, os conjuntos de predicados que podem estar associados a algum sujeito. O processo de identificação de CSs gera uma base categorizada, como definido abaixo:

Definição 1: Uma base RDF categorizada B_c de uma base RDF B é definida como um par (S, M), onde: S é um conjunto de CSs, e M é um mapeamento de S para um conjunto de sujeitos, tal que $M(c)$ = $\{\ s \mid pred(s)\ =\ c,\ c \in S\}$. □

A partir da base RDF do Exemplo 1, tem-se CS_1 = $\{p1, p2, p3, p4\}$, CS_2 = $\{p2, p5\}$, CS_3 = $\{p2, p5, p7\}$ e CS_4 = $\{p8\}$. Logo, o conjunto S = $\{CS_1, CS_2, CS_3, CS_4\}$, e $M(CS_1)$ = $\{s1, s2\}$, $M(CS_2)$ = $\{s3, s4\}$, $M(CS_3)$ = $\{s5\}$ e $M(CS_4)$ = $\{s6\}$.

Passo 2: Atribuição de rótulos

O passo seguinte consiste da atribuição de rótulos (Linha 3). O resultado do processo é uma função *label*, que mapeia cada CS c em S para um rótulo, que será utilizado como o nome da tabela de entidade que comportará os sujeitos em $M(c)$. A escolha do rótulo leva em consideração 3 características. A primeira é a existência do predicado *type*. Caso ele exista, o predicado *type* com maior frequência no CS, e superior a uma frequência mínima T_{inf}, será selecionado como rótulo. Por exemplo, o rótulo atribuído ao CS_1 é *Pessoa*, uma vez que $M(CS_1)$ = $\{s1, s2\}$ e ambos possuem o predicado *type* com valor *Pessoa*. A segunda característica considerada é o relacionamento entre CSs. A atribuição do rótulo se dá de acordo com o predicado que tem o CS em questão como objeto. Por exemplo, suponha que s_3 e s_4 não possuíssem o predicado *type*. Neste caso, como ambos são objetos da propriedade *veiculo*, este rótulo seria escolhido como $label(CS_2)$. Caso os dois casos anteriores não puderem ser aplicados, o CS recebe o rótulo da sua propriedade mais frequente, realizando apenas a remoção do prefixo. Por

exemplo `http://www.exemplo.br/observacao` é reduzido para `observacao`.

Passo 3: Junção de CSs
O passo de junção de CSs (Linha 4) tem por propósito diminuir a quantidade total de CSs, agrupando dados similares. A primeira junção é de CSs de mesmo rótulo que foram atribuídos pela regra de predicado *type*. No exemplo corrente, CS_2 e CS_3 são agrupados, uma vez que ambos possuem o rótulo *Automotor*, atribuídos devido ao predicado *type*. O parâmetro de configuração T_{inf} é utilizado para agrupar CSs referenciados pelo mesmo predicado ou que possuem similaridade superior a este limiar [Pham et al. 2015]. Como resultado da junção de dois CSs, cs_x e cs_y, o cs_y é removido do conjunto S, e os predicados e sujeitos de cs_y passam a pertencer a cs_x. Ou seja, $pred(cs_x)$ recebe $(pred(cs_x) \cup pred(cs_y))$ e $M(cs_x)$ recebe $(M(cs_x) \cup M(cs_y))$. No exemplo corrente, após a junção, $S = \{CS_1, CS_2, CS_4\}$, $pred(CS_2) = \{p2, p5, p7\}$ e $M(CS_2) = \{s3, s4, s5\}$.

Passo 4: Geração de mapeamento
O resultado final do processo de junção é armazenado em uma estrutura chamada *map*, criada com a chamada da função *geraMapeamento* (Linha 5). É a partir desta estrutura que os passos de filtragem e geração da base (Linhas 6-8) são executados.

Definição 2: Considere um conjunto de tipos de literais L e uma base RDF categorizada $B_c = (S, M)$, com função de atribuição de rótulos $label$. A estrutura map é um vetor multidimensional, tal que o elemento $map[c][p][k][t]$ contém uma tupla *(nomeTabela, nomeColuna, flg_multivalorado, flg_emOverflowE)*, onde: c é um CS em S; p é um predicado que pertence a c; k contém a constante `lit` ou a constante `fk`; se k = `lit` então t é um tipo em L; caso k = `fk`, t contém $label(c')$ para algum c' em S. Na tupla associada: *nomeTabela* contém o nome da tabela na qual o predicado p de c é armazenado e é obtido pela concatenação de $label(c) +' RDF'$; *nomeColuna* é o atributo em *nomeTabela* que contém p e é obtido pela remoção do prefixo de p; *flg_multivalorado* contém *true* se o predicado p é multivalorado e *false*, caso contrário; *flg_emOverflowE* contém *true* se o predicado p está na tabela de overflow específico de *nomeTabela* e *false*, caso contrário. □

Para exemplificar, um dos elementos da estrutura map no exemplo corrente é $map[CS_1][p1][\texttt{lit}][string] = (' PessoaRDF',' nome', false, false)$, uma vez que o CS_1 contém o predicado $p1$, com valor literal do tipo string, que será armazenado na tabela *PessoaRDF* na coluna *nome*. Além disso, este predicado não é multivalorado (*flg_multivalorado* é $false$) e nem é armazenado na tabela de overflow específico (*flg_emOverflowE* é *false*). Para a geração da estrutura, a base RDF B é percorrida a fim de criar, para cada CS c, 3 conjuntos: *multiValued(c)*, que contém os predicados multivalorados de c; *literalPred(c)* e *linkPred(c)*, que contêm os predicados com valores literais e com ligações para outros CSs, respectivamente, juntamente com seus tipos associados. No exemplo adotado, para CS_1 são criadas as seguintes listas: *multiValued(CS_1) = {$p3$}*, *literalPred(CS_1) = {($p1$,string),($p2$,string),($p3$,string)}* e *linkPred(CS_1) = {($p4$,CS_2)}*. A partir destes conjuntos, a estrutura map é preenchida. Os elementos da estrutura map associados a CS_1 resultantes são:

map[CS_1][$p1$][`lit`][string] = ('PessoaRDF', 'nome', false, false),
map[CS_1][$p2$][`lit`][string] = ('PessoaRDF', 'type', false, false),
map[CS_1][$p3$][`lit`][string] = ('telefoneMultivalueRDF', 'telefone', true, false),
map[CS_1][$p4$][`fk`][CS_2] = ('PessoaRDF', 'fk_veiculo', false, false).

Observe que o nome da tabela é alterado quando o atributo é multivalorado, sendo de-

finido pela concatenação do nome do atributo com 'MultivalueRDF'. De forma similar, atributos que são `fk`, tem o seu nome acrescido do prefixo 'fk_'. Além disso, o valor *false* é atribuído para *flg_emOverflowE* em todos os elementos. Isso porque a transferência de atributos e tabelas para o overflow é feita no passo de filtragem, que corresponde ao processo de refinamento do esquema.

Passo 5: Filtragem
No passo de filtragem (Linha 6), tabelas menos significativas são movidas para o overflow geral, e atributos menos significativos para o overflow específico respectivo. Inicialmente é verificado se cada CS possui a quantidade mínima de sujeitos (min_t), que é um parâmetro de configuração. Caso não possua, ele é migrado para a tabela de overflow geral. No exemplo, se min_t for maior ou igual a 2, o CS_4 é movido para o *Overflow*, uma vez que $|M(CS_4)|$ = 1. Como resultado, a estrutura *map* é alterada, removendo a entrada $map[CS_4][p8]$[`lit`][string] e inserindo a entrada $map[CS_{over}][p8]$[`lit`][string] com o valor *('Overflow', 'observacao', false, false)*. Além disso, os sujeitos em $M(CS_4)$ passam a pertencer a $M(CS_{over})$, onde CS_{over} é o CS introduzido para ser associado ao overflow geral. A filtragem contém ainda a verificação se a quantidade de CSs é maior do que o máximo de tabelas permitido (Ub_{tbl}). Caso seja maior, os CSs com menor quantidade de sujeitos também são migrados para o *Overflow*. A única exceção é feita para tabelas dimensionais, pelo fato delas serem frequentemente referenciadas por outras tabelas. No exemplo, caso $Ub_{tbl} = 1$ e $S = (CS_1, CS_2)$, como $|M(CS_1)| < |M(CS_2)|$ e CS_1 não é uma tabela dimensional, ela seria migrada para o *Overflow*.

Por fim, é realizado o processo de minimização de propriedades infrequentes a fim de que as propriedades com baixa frequência sejam eliminadas das tabelas de entidades e colocadas no overflow específico, uma vez que elas ocasionariam colunas com muitos valores nulos. O parâmetro T_{inf} é utilizado para determinar se um predicado é infrequente. Seguindo o exemplo, suponha que T_{inf} seja 0.4. No CS_2 o predicado $p7$ está presente em apenas um sujeito ($s5$) e $|M(CS_2)| = 3$. Como a presença é de $1/3 = 0.33 < T_{inf}$, $p7$ é movido para o overflow específico de CS_2. Logo, a estrutura *map* é alterada para conter o valor *true* para o *flg_emOverflowE*. Assim, *map[CS_2][p7][`lit`][int]* passa o conter o valor *('Overflow_AutomotorRDF', 'cod_modelo', false, true)*.

Passo 6: Criação de tabelas
Baseado na estrutura map, as tabelas para armazenar as triplas RDF são geradas. Cada tabela de entidade é criada contendo todos os atributos que *flg_multivalorado* e *flg_emOverflowE* possuem valor *false*. Assim, no exemplo são criadas as seguintes tabelas: *PessoaRDF (OID, nome, type, fk_veiculo)*, onde *OID* é um identificador numérico criado pelo sistema, e *AutomotorRDF (OID, modelo, type)*. Cada tabela de entidade tem uma tabela de overflow específico associada: *Overflow_PessoaRDF* e *Overflow_AutomotorRDF*. Além disso, é criada uma tabela para cada entrada que possui o *flg_multivalorado* com valor *true*. No exemplo, é criada a tabela *telefoneMultivalueRDF*. Estas tabelas são ilustradas na Figura 2. É importante observar que todas as informações necessárias para a criação das tabelas encontram-se na estrutura map.

A estrutura map contém também informações para a inserção de dados nas tabelas de metadados *TB_FullPredicate* e *TB_DatabaseSchema*. A tabela *TB_FullPredicate* tem como objetivo associar as IRIs completas dos predicados (p_i) com os CSs que os possuem e as tabelas e atributos nos quais estão armazenados. Assim, para cada entrada

na estrutura $map[c][p][k][t]$ com valor *(nomeTabela, nomeColuna, flg_multivalorado, flg_emOverflowE)*, é gerada uma linha na tabela *TB_FullPredicate*, com os seguintes valores: *cs_identifier:=* c; *tableName:= null* se *flg_emOverflowE* for *true* e *nomeTabela*, caso contrário; *overflowTable:= null* se *flg_emOverflowE* for *false* e *nomeTabela*, caso contrário; *columnName:= nomeColuna*; e *predicate:=* p. A Figura 2 apresenta a tabela *TB_FullPredicate* do exemplo corrente.

A tabela *TB_DatabaseSchema* armazena informações sobre o mapeamento da base RDF para o relacional. Para cada CS ela mantém: o identificador do CS (*cs_identifier*), nome do predicado sem o prefixo (*propertyName*), tipo do objeto (*valueType*), nome da tabela em que se encontra armazenado o predicado (*tableName*) e o nome do atributo (*tableAttribute*). O *valueType*, quando não for *literal*, contém o identificador do CS referenciado. Considere a tabela *TB_DatabaseSchema* da Figura 2. O *valueType* do predicado *fk_veiculo* de CS_1 é CS_2. Isso significa que o atributo *fk_veiculo* da tabela *PessoaRDF* é uma chave estrangeira que referencia a tabela *AutomotorRDF*.

Passo 7: Carregamento de Dados
Cada tripla da base RDF B só pode ser inserida em uma das quatro tabelas: *Overflow*, tabela de entidade, tabela de overflow específico associada ou tabela de propriedade multivalorada. Estas informações encontram-se na estrutura map. Além disso, para cada CS c, $M(c)$ contém o conjunto de sujeitos que pertencem a c. Portanto, para cada tripla (s, p, o) determina-se a qual $M(c)$ o sujeito s pertence, cria-se um novo OID (caso ele ainda não exista) e insere-se o valor na tabela apropriada, de acordo com a estrutura map. Além disso, a tabela de metadados *TB_Subj_OID* associa as IRIs completas do sujeitos (s_i) aos seus identificadores (OIDs), como mostrado na Figura 2.

Com isso, vale salientar as diferenças do AORR nos passos que se assemelham aos do ERSR. Dado que o AORR tem o intuito de ser utilizado como *backend* de armazenamento, a atribuição por propriedade discriminativa e a junção por ancestral ontológico do ERSR não foram utilizadas no AORR. Já no passo de filtragem a diferenciação se dá por não existir OverflowE no ERSR, ou seja, o que no AORR faz menção a OverflowE, no ERSR seria o OverflowG.

3.2. Atualização da Base

A existência das tabelas de metadados permite que o AORR seja capaz de realizar atualizações incrementais da base. A inserção de novas triplas usa a seguinte estratégia geral: sempre que possível a inserção é realizada em uma tabela de entidade ou em seu overflow específico. Considere a inserção de uma tripla (s, p, o). É possível determinar a tabela de entidade a qual s pertence em 2 casos. No primeiro, s já faz parte da base, o que pode ser determinado com uma busca na tabela de metadados *TB_Subj_OID*, e o valor do atributo *tableName* não é *Overflow*. Caso o sujeito não seja encontrado, um novo identificador OID é gerado e o sujeito é inserido na tabela *TB_Subj_OID*. Novos sujeitos só serão inseridos em uma tabela de entidade pelo segundo caso, que refere-se à inserção de triplas com o predicado *type*. É importante observar que as tabelas de entidades podem armazenar dados de um conjunto de tipos associados similares, mas cada *type* está associado a uma única tabela. Assim, uma tripla $(s, type, o)$ é armazenada em uma tabela de entidade T se T possuir uma coluna *type* com pelo menos uma linha com valor igual a o. Caso esta tabela não exista, a tripla é armazenada no overflow geral. Uma situação importante a ser observada é que outros predicados do sujeito s podem ter sido anteriormente inseridos na

base e, por não ter sido possível determinar sua tabela de entidade associada, elas foram armazenadas no overflow geral. Assim, a inserção de uma tripla $(s, type, o)$ pode resultar na migração de triplas do overflow geral para uma tabela de entidade.

A inserção de uma tripla (s, p, o) em uma tabela de entidade T, associada a um CS c, segue os seguintes passos. Primeiro, é verificado se c possui o predicado p. Para isso, a tabela de metadados *TB_FullPredicate* é pesquisada. Caso não exista, a tripla é inserida no overflow específico de T e as tabelas de metadados *TB_FullPredicate* e *TB_DatabaseSchema* são atualizadas para registrar a existência de um novo predicado no CS c. Se o predicado p já existir e for multivalorado, a tripla é inserida na tabela associada a este atributo, que está indicada pelo *tableName* da tabela *TB_FullPredicate*. Se p não for multivalorado, obtém-se (ou cria-se) a linha na tabela de entidade e atribui-se o valor o à coluna correspondente. Todavia, é possível que o atributo já possua um valor. Isso ocorre quando esperava-se que p fosse um atributo monovalorado. Neste caso, a tripla é inserida na tabela de overflow específico e as tabelas de metadados são atualizadas para registrar a existência do predicado tanto na tabela de entidade como no seu overflow.

Exemplo 3: Considere a inserção da tripla $(s5, p5,' carro\ popular')$ na base ilustrada na Figura 2. Primeiro, é realizada uma busca na tabela de metadados *TB_Subj_OID* para verificar a existência do sujeito $s5$. Ele é encontrado com OID 5 e com o valor do atributo *tableName* igual a *AutomotorRDF*. Na sequência, é realizada uma busca na tabela *TB_FullPredicate* pelo predicado $p5$ com o CS associado à tabela AutomotorRDF. O predicado é encontrado, de onde obtém-se que o nome da coluna correspondente a $p5$ denomina-se *modelo*. A linha da tabela *AutomotorRDF* com OID 5 é então recuperada para preencher o valor do atributo *modelo*. No entanto, esta linha já possui este campo preenchido. Assim, a tripla *(5, modelo, carro popular)* é inserido na tabela *Overflow_AutomotorRDF*. Para registrar a existência de $p5$ tanto na tabela de entidade como no seu overflow, as tabelas *TB_FullPredicate* e *TB_DatabaseSchema* são atualizadas da seguinte forma. Na tabela *TB_FullPredicate*, a linha referente ao predicado $p5$ terá tanto o atributo *tableName* preenchido com *AutomotorRDF*, como o atributo *overflowTable* preenchido com *Overflow_AutomotorRDF*. Na tabela *TB_DatabaseSchema*, uma linha será inserida com os valores *(CS2, modelo, literal, Overflow_AutomotorRDF, modelo)*. □

3.3. Processamento de Consultas

Esta seção exemplifica a utilização das tabelas de metadados na tradução de consultas SPARQL para consultas SQL. A tabela *TB_FullPredicate* é utilizada para encontrar o atributo referente ao predicado da consulta, enquanto a tabela *TB_DatabaseSchema* é utilizada para identificar relacionamentos entre as tabelas de entidades e seguir com a busca pelo padrão de triplas. Considere a consulta SPARQL da Figura 3(a) executada sobre a base ilustrada na Figura 2 após a inserção da tripla $(s5, p5,' carro\ popular')$ do Exemplo 3. A consulta SQL resultante da tradução está ilustrada na Figura 3(b). Na consulta, a tabela *TB_DatabaseSchema* é utilizada para determinar quais CSs possuem os dois predicados, *modelo* e *cod_modelo* (referentes a $p5$ e $p7$), pois ambos estão associados ao mesmo sujeito $?v$ na consulta. Determina-se que apenas CS_2 satisfaz esta condição e que o o predicado *modelo* está presente tanto na tabela *AutomotorRDF*, como em seu overflow *Overflow_AutomotorRDF*, enquanto o predicado *cod_modelo* está presente apenas no *Overflow_AutomotorRDF*. Sendo assim, duas subconsultas são geradas, uma para obter o atributo *modelo* da tabela *AutomotorRDF* e *cod_modelo* da tabela *Over-*

```
SELECT ?m ?c WHERE {
    ?v p5 ?m .
    ?v p7 ?c .
}
```

(a)

```
SELECT v.modeloXQ9G2 AS m,
    v.cod_modelo0VF65 AS c
FROM (SELECT a1.OID,
        a1.modelo AS modeloXQ9G2,
        o1.obj AS cod_modelo0VF65
    FROM AutomotorRDF a1,
        Overflow_AutomotorRDF o1
    WHERE a1.modelo IS NOT NULL
        AND o1.pred = 'cod_modelo'
        AND o1.obj IS NOT NULL
UNION ALL
SELECT o1.subj,
    o1.obj AS modeloXQ9G,
    o2.obj AS cod_modelo0VF65
FROM Overflow_AutomotorRDF o1,
    Overflow_AutomotorRDF o2
WHERE o1.subj = o2.subj
    AND o1.pred = 'modelo'
    AND o2.pred = 'cod_modelo'
    AND o1.obj IS NOT NULL
    AND o2.obj IS NOT NULL) v
```

(b)

Figura 3. Consulta SPARQL traduzida em consulta SQL

flow_AutomotorRDF e outra para obter os dois atributos de *Overflow_AutomotorRDF*. Elas são unidas com o operador UNION ALL para produzir o resultado final.

Esta tradução de consulta mostra que o volume de dados armazenados em tabelas de overflow podem ter um grande impacto no desempenho das consultas. Como no ERSR, o trabalho sobre o qual o processo de extração de esquema do AORR foi inspirado, não considera a existência de overflow específicos, mas apenas um único overflow geral, a consulta acima necessitaria fazer duas junções com tabelas potencialmente volumosas. O impacto da existência de tabelas de overflow específico sobre o desempenho das consultas é avaliado no experimento relatado na próxima seção.

4. Análise Experimental

As tabelas de overflow específico dividem a carga do overflow geral de forma a diminuir o volume de dados envolvidos no processamento das consultas. O experimento relatado nesta seção avalia o impacto desta estratégia no tempo de processamento de consultas, utilizando a base RDF Peel[4], que possui 276160 triplas e 45 MBytes. As características da base relacional gerada pelo AORR estão descritas na Tabela 1, com um volume total de 29 MBytes. O sistema foi implementado na linguagem Python 2.7, utilizando o SGBD MySQL v 14.14. Os experimentos foram executados em um computador com processador Intel i7-4710HQ, 2.50GHz e 2GB de memória, com S.O Linux 3.13.0. Os parâmetros de entrada de quantidade máxima de tabelas, quantidade mínima de registros e frequência mínima utilizados foram 7, 1000 e 0.1 respectivamente. Tais parâmetros foram adotados para que todos os passos do algoritmo fossem executados e algumas tabelas fossem migradas para o OverflowG pelo processo de filtragem.

Foram elaboradas 5 consultas (C1,...,C5) que exploram diferentes situações nas quais as tabelas de overflow são utilizadas [Pauluk et al. 2018]. Tais consultas foram traduzidas de SPARQL para SQL como detalhado na Seção 3.3. As consultas foram executadas sobre duas bases, sendo que uma delas possui tabelas de overflow específico (com OverflowE) e a outra apenas o overflow geral (sem OverflowE). Foram utilizadas bases de dados com 6 tamanhos: a base Peel original, e 5 outras geradas a partir dela com incrementos de 5000, 10000, 20000, 40000 e 80000 triplas nas tabelas de overflow, para cada predicado utilizado nas consultas. Por exemplo, para uma consulta que envolve 2 predicados associados a um mesmo sujeito, o incremento de 5000 gera 5000 novos sujeitos e triplas suficientes para gerar 5000 novas linhas em tabelas de overflow para cada predicado, totalizando um incremento de 10.000 linhas de overflow. Na discussão abaixo a base com overflow específico e incremento N será chamada de R^N_{AORR} e a base sem overflow específico é denotada por R^N_{ERSR} e contém uma única tabela de overflow geral chamada de *OverflowG*. A Figura 4 apresenta os resultados para as cinco consultas nas

[4] `http://dbtune.org/bbc/peel/`

Tabela 1. Base relacional gerada pelo AORR

Tabela	# Linhas	Volume (MB)
fk_engineerMultivalueRDF	3801	0.16
fk_performedMultivalueRDF	11869	0.49
fk_sub_eventMultivalueRDF	30062	1.55
instrumentMultivalueRDF	8926	0.43
MusicalWorkRDF	18273	2.93
PerformanceRDF	28631	4.13
PersonRDF	10611	1.76
RecordingRDF	3976	1.66
SignalRDF	5947	1.7
SoundRDF	3976	0.4
TransmissionRDF	3976	1.65

Tabela	# Linhas	Volume (MB)
Overflow	7417	0.45
Overflow_MusicalWorkRDF	938	1.29
Overflow_PerformanceRDF	1330	0.14
Overflow_PersonRDF	7705	1.78
Overflow_RecordingRDF	501	0.08
Overflow_SignalRDF	782	0.1
Overflow_SoundRDF	0	0.03
Overflow_TransmissionRDF	1800	0.21
TB_DatabaseSchema	55	0.02
TB_FullPredicate	45	0.02
TB_Subj_OID	76894	7.7

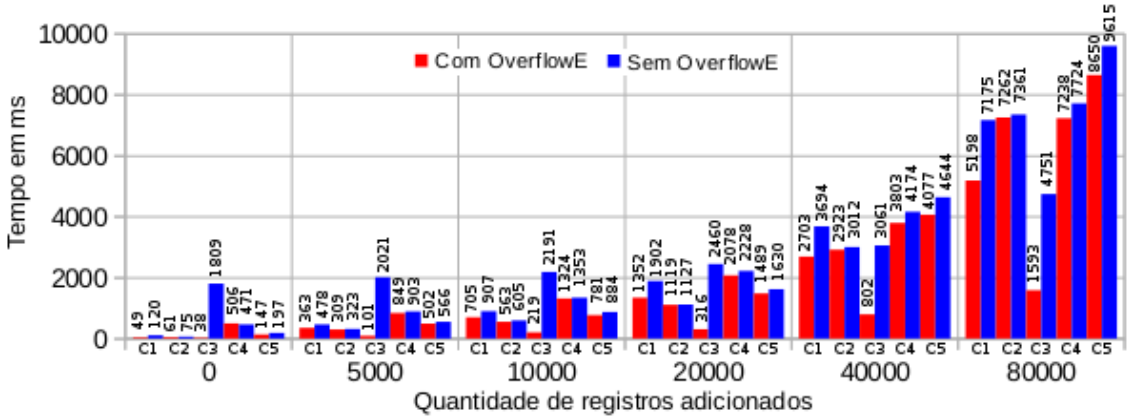


Figura 4. Resultado das consultas para cada incremento em milissegundos

bases R_{AORR} e R_{ERSR}, para cada tamanho de base. Os tempos são reportados em milissegundos e consideram apenas o tempo de processamento da consulta, sem considerar o tempo de tradução. O único índice criado sobre as tabelas foi sobre o atributo OID.

A consulta na qual o AORR apresentou maior ganho foi a C3 com um tempo de processamento 198% menor para a base R^{80000}_{AORR} comparada à base R^{80000}_{ERSR}. Isso se deve ao fato da consulta envolver apenas tabelas de overflow. Na base R^{80000}_{ERSR}, a consulta executa duas auto-junções sobre o *OverflowG*, que possui aproximadamente 820.000 registros. Em contrapartida, na base R^{80000}_{AORR} é executada uma auto-junção de *Overflow_PersonRDF*, que possui 160.000 registros aproximadamente, e uma junção com *Overflow_RecordingRDF*, com 80.000 registros. Já a consulta C1 busca por dois predicados, ambos contidos tanto na tabela *SignalRDF* como no *Overflow_SignalRDF*. Com isso, a C1 tem 4 subconsultas, precisando realizar três operações UNION ALL, para que todas as combinações sejam contempladas. Como pode ser percebido, o desempenho nas bases R_{AORR} para C1 aumentam proporcionalmente em relação aos incrementos realizados. Já na base R_{ERSR}, o crescimento é maior, dado que a tabela *OverflowG* cresce em uma proporção maior que *Overflow_SignalRDF*. Além disso, como ambos os atributos estão em tabelas de overflow, uma auto-junção sobre *OverflowG* é bem mais custosa do que sobre *Overflow_SignalRDF*. A consulta C4 realiza a busca por apenas um predicado, sendo que ele está contido em 9 tabelas, sendo 2 overflow específicos e 1 overflow geral na base R_{AORR}. Já na base R_{ERSR} a consulta é realizada sobre 7 tabelas, sendo uma delas o *OverflowG*. Como no cenário sem incremento as tabelas de overflow são relativamente pequenas, consultar 2 tabelas de overflow específico no R_{AORR} foi mais custoso do que na base R_{ERSR}. Todavia, já no incremento de 5000, a consulta sobre o R_{AORR} é mais vantajosa do que sobre o R_{ERSR}, dado que o *OverflowG* cresce mais do que as tabelas

de overflow específico. Para a C5, como existem 6 junções em cada subconsulta e duas subconsultas para um UNION ALL, os resultados mostram que a consulta sobre a base R_{AORR} em relação à base R_{ERSR} começa a apresentar um ganho a partir do incremento de 40000. Logo, não houve grande impacto nas consultas de menor incremento.

5. Conclusão

Esse artigo apresentou o AORR, uma abordagem para armazenar dados RDF em um SGBDR e habilitar o processamento otimizado de consultas de casamento básico de grafos. O AORR dá suporte à atualização incremental da base e à tradução de consultas SPARQL em consultas SQL. Os resultados apresentados mostram que consultas realizadas sobre a base gerada pelo AORR apresentaram melhor desempenho do que uma simulação da base gerada pelo ERSR [Pham et al. 2015]. Tal ganho se deu principalmente devido às tabelas de overflow específico. As consultas realizadas sobre as bases de teste foram geradas a partir de um processo de tradução de consulta SPARQL em consulta SQL, que é possível graças às tabelas de metadados propostas pelo AORR. Dentre os trabalhos futuros podem ser citados: desenvolvimento de um processo de migração de triplas na tabela de overflow geral para tabelas de entidades; geração automática dos parâmetros de configuração usados para a geração do esquema relacional; exploração de recursos de otimização do SGBDR, como indexação e caching; e execução de experimentos com outras bases de dados e *benchmarks*.

Referências

Abadi, D. J., Marcus, A., Madden, S. R., and Hollenbach, K. (2007). Scalable semantic web data management using vertical partitioning. In *VLDB*, pages 411–422.

Aluç, G., Ozsu, M. T., and Daudjee, K. (2014). Workload matters: Why rdf databases need a new design. *Proceedings of the VLDB Endowment*, 7(10):837–840.

Bornea, M., Dolby, J., Kementsietsidis, A., Srinivas, K., Dantressangle, P., Udrea, O., and Bhattacharjee, B. (2013). Building an efficient rdf store over a relational database. In *ACM SIGMOD*.

He, L., Shao, B., Li, Y., Xia, H., Xiao, Y., Chen, E., and Chen, L. J. (2017). Stylus: A strongly-typed store for serving massive rdf data. *Proc. VLDB Endow.*, 11(2):203–216.

MahmoudiNasab, H. and Sakr, S. (2010). An experimental evaluation of relational rdf storage and querying techniques. In *Proc. of DASFAA*, pages 215–226.

Pauluk, J. G., Duarte, M. M. G., Prado, R. L., and Hara, C. S. (2018). Processamento de Consultas SPARQL em uma Base Relacional de Entidades. In *SBBD - Short Papers*.

Penteado, R. R. M., Schroeder, R., and Hara, C. S. (2015). Exploração de grafos RDF com distribuição controlada. In *Anais do XXX SBBD - Short Papers*, pages 69–74.

Pham, M.-D., Passing, L., Erling, O., and Boncz, P. (2015). Deriving an emergent relational schema from rdf data. *Proc. of the 24th WWW Conf.*, pages 864–874.

Ramunajam, S., Gupta, A., Khan, L., Seida, S., and Thurasaisingham, B. (2009). R2d: Extracting relational structure from rdf stores. In *Proc. of the IEEE/ACM WIC*, pages 361–366.

Scabora, L. C., Oliveira, P. H., Kaster, D. S., Traina, A. J. M., and Traina-Jr, C. (2017). Relational graph data management on the edge: Grouping vertices' neighborhood with edge-k. In *Anais do XXXII SBBD*, pages 124–135.

Zeng, K., Yang, J., Wang, H., Shao, B., and Wang, Z. (2013). A distributed graph engine for web scale rdf data. *Proc. of the VLDB Endowment*, 6(4):265–276.

REALM: Um Framework Computacional para Investigar os Impactos de Pesquisas Através de Métricas Alternativas

Luís Fernando Monsores Passos Maia[1], **Jonice Oliveira**[1]

[1]Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brasil

luisfmpm@ufrj.br, jonice@dcc.ufrj.br

Abstract. *In some emergency scenarios or undefined domains, more collaboration among specialists is required. For instance, we can mention the Zika virus, whose epidemic potential became evident in 2014. In Brazil, it had a high occurrence rate, affecting thousands of people and causing overcrowding of public and private emergency services. The social media has been used as the primary alternative to exchange information and create scientific knowledge. The researchers and physicians use social media to communicate their discoveries - abdicating of official and scientific publications - because they needed a faster and proactive way to exchange knowledge. This scenario demands mechanisms to identify experts and to recognize how citizens interpret the efficiency of professionals and their efforts to find solutions. For this purpose, we created a computational framework to identify the social reputation of a researcher or a study, based on alternative impact metrics (altmetrics). To evaluate this framework, we did a Proof of Concept in the Zika context.*

Resumo. *Em alguns cenários de emergência ou em que não há solução conhecida, é necessário mais colaboração entre os especialistas. A exemplo disso tivemos a epidemia de Zika virus, que entre os anos de 2014 e 2016 ganhou destaque internacional, tanto pela sua proporção epidêmica quanto pelas suas consequências. No Brasil, o surto rapidamente evoluiu para uma situação de emergência de saúde pública, exigindo a cooperação dos especialistas e celeridade de resposta. A demanda por resultados rápidos fez com que médicos e pesquisadores publicassem suas descobertas nas mídias sociais, abdicando das publicações científicas oficiais. Cenários como este demandam mecanismos para identificar especialistas e como a população interpreta as soluções por eles criadas. Para este fim, desenvolvemos um framework computacional para medir a reputação social de cientistas e suas pesquisas, com base em métricas de impacto alternativas (altmetrics). Para avaliar o framework, realizamos uma prova de conceito com especialistas no domínio do Zika vírus.*

1. Introdução

Em alguns cenários de emergência ou em que não há solução criada, torna-se necessário uma maior colaboração entre os especialistas. Como exemplo, pode-se citar a epidemia de Zika Virus (ZIKV), cujo potencial epidêmico tornou-se evidente em 2014. No Brasil, a partir de 2015, o surto apresentou uma alta taxa de ocorrências, afetando milhares de pessoas e causando superlotação dos serviços de emergência públicos e privados, embora não tenha sido medido por um sistema de notificação oficial [Zanluca et al. 2015].

Casos de microcefalia e alterações neurológicas em recém-nascidos ocorreram em Pernambuco e em outros estados do Nordeste e, a posteriori, no Sudeste do país, levando o governo brasileiro a declarar estado de Emergência em Saúde Pública de Importância Nacional em novembro de 2015, e posteriormente pela Organização Mundial de Saúde, em 01/02/2016, uma Emergência de Saúde Pública de Importância Internacional (*Public Health Emergency of International Concern*, PHEIC) [Oliveira and Vasconcelos 2016].

Pela urgência de respostas, a ciência se apressou nas investigações. Para garantir que a comunidade científica internacional tivesse condições de tornar públicos os resultados e discussões sobre o tema, foram adotados procedimentos mais rápidos para a aprovação e publicação de artigos sobre o assunto, os chamados *fasttracks*. A demanda pela divulgação de resultados rápidos fez também com que muitos pesquisadores começassem a mostrar seus resultados nas mídias sociais, não esperando o tempo das publicações convencionais [Harmon 2016; McNeil Jr 2016]. Cenários como esse oferecem uma excelente oportunidade para verificar a reputação de especialistas, sua produção científica e como a população interpreta as soluções por eles criadas.

Uma forma eficaz de analisar a produção científica é através de métricas de Análise de Redes Sociais (ARS) e do mapeamento de redes de colaboração científica - também conhecidas como Redes Sociais Científicas (RSC) -, já que atualmente, a colaboração constitui uma característica intrínseca da ciência moderna. Deste modo, a coautoria se apresenta como um importante indicador de colaboração científica na compreensão de diversos fatores relacionados à colaboração entre especialistas [Maia et al. 2018].

Além disso, novas abordagens para avaliar o impacto científico vêm ganhando espaço na medida em que os cientistas mudam seus comportamentos de pesquisa e divulgação para a web [Priem and Hemminger 2010]. Em função disso, métricas alternativas de impacto científico baseadas em mídias sociais estão sendo desenvolvidas e testadas [Priem 2013]. Este novo tipo de medição, também conhecido como Altmetria, consiste em métricas alternativas (*altmetrics*) que permitem mapear a correlação entre os pesquisadores e a sociedade, que vem se estreitando cada vez mais através da troca de experiências, avaliações e conteúdos em mídias sociais, *wikis*, blogs e microblogs, sites de notícias, fóruns de discussão, Redes Sociais On-line (RSO), etc [Bornmann 2014].

Para Priem et al. (2012) a Altmetria pode ser utilizada como uma ferramenta para auxiliar os pesquisadores, não apenas em seus campos de atuação, mas para maximizar a influência e os impactos de suas pesquisas, de modo que seja possível medir sua relevância e contextualizá-la em um universo cada vez mais concorrido de trabalhos científicos.

Deste modo, este trabalho tem como proposta a criação de um *framework* para medir os impactos da ciência na atualidade, tendo em vista entender a representatividade e reconhecimento dos pesquisadores perante a sociedade. O *framework* computacional, denominado REALM (*Researcher Evaluation ALternative Metrics*), visa identificar a reputação social de pesquisadores e suas pesquisas, baseando-se em métricas de impacto alternativas, também conhecidas como *altmetrics* [Priem and Hemminger 2010]. O *framework* foi aplicado no cenário da Zika, onde as questões de pesquisa levantadas foram: “Quem são os pesquisadores mais influentes academicamente?"e "Quem são os pesquisadores com maior inserção na população, possuindo um alto impacto social?”. Os resultados obtidos foram avaliados por especialistas no domínio do ZIKV.

2. O *framework* computacional REALM

O *framework* REALM consiste em uma infraestrutura de *software* que permite a coleta e fusão de dados de publicações em bases de dados indexadas e em mídias sociais com o propósito de extrair e correlacionar diferentes grupos de métricas (produtividade, impacto acadêmico e impacto social). A partir disso é possível identificar, com maior precisão que outros métodos tradicionais (exemplo: número de citações ou h-index), quem são os pesquisadores e/ou grupos de destaque em tópicos de interesse e como esses pesquisadores estão colaborando para engendrar soluções e novas tecnologias. O REALM divide-se em quatro módulos: (a) Coleta e tratamento de dados de publicações acadêmicas; (b) Coleta e tratamento de dados de mídias sociais; (c) Análise do impacto acadêmico; (d) Análise do impacto social.

2.1. Módulo de coleta e tratamento de dados de publicações acadêmicas

Este módulo é responsável pela recuperação de dados de publicações em bases de dados indexadas (exemplo: PubMed[1] e Web of Science[2]) para construção de RSC de coautoria com base em áreas/tópicos de interesse específicos (exemplo: Zika, Dengue e Chikungunya). O módulo opera extraindo das publicações dados como título, nome dos autores, afiliações, data de publicação, identificador do artigo, entre outros. A partir disso ocorrem a separação e tratamento desse conjunto de dados para uma tabela contendo a formatação de grafo, onde os nós representam os pesquisadores e as arestas representam suas publicações em comum. As principais operações realizadas por este módulo são: (i) associação de dois nós (autores), com base no título de uma publicação, caracterizando uma aresta. (ii) Remoção de arestas sem nós associados. (iii) Junção dos nós de autores associados no item (i) em uma única coluna, resultando em um *array* de autores separados por vírgula. (iv) Remoção de itens duplicados. (v) Criação das *labels* dos autores associadas ao *array* do item (iii), resultando na coluna da coautoria. (vi) Verificação do número de vezes que a combinação do item (v) se repete para posterior atribuição de pesos às arestas. (vii) Atribuição de identificadores a cada nó e aresta, possibilitando a leitura e armazenamento dos dados da RSC no banco de dados para posterior visualização do grafo de coautoria e extração de métricas de impacto acadêmico.

2.2. Módulo de coleta e tratamento de dados de mídias sociais

Este módulo é responsável pela coleta, pré-procesamento e triplificação de dados de publicações em mídias sociais como jornais on-line (sites de notícias), blogs científicos, fóruns de discussão e RSO como Facebook, Twitter, Google+, entre outras. Ele corresponde a uma implementação do processo *Extract, Transform, Load* (ETL) descrito em [Maia and Yagui 2017], onde os autores analisaram a repercussão do ZIKV em mídias sociais por ocasião das Olimpíadas Rio 2016. A coleta desses dados ocorre a partir de uma implementação da API do serviço Webhose.io[3], que permite o monitoramento de mídias sociais em tempo real e a coleta de publicações de modo automático 24h/dia. A configuração de '*queries*' específicas no código possibilita reduzir o escopo da coleta para tópicos de interesse específicos, extraindo apenas publicações relacionadas a determinados temas (exemplo: Zika, Dengue e Chikungunya). Após a coleta os dados (não estruturados)

[1]https://www.ncbi.nlm.nih.gov/pubmed/

[2]https://webofknowledge.com/

[3]https://webhose.io/

dessas publicações são convertidos para o formato semi-estruturado (JSON/XML) e enviados para um componente do módulo onde ocorre seu pré-procesamento. Neste ponto são extraídos campos/termos da publicação, tais como URI, título, texto, autor, país, domínio, data, idioma, compartilhamentos em RSO, entre outros, de modo que, a partir desses dados torna-se possível a extração dos índices altmétricos e, consequentemente, das métricas de impacto social. A seguir ocorre a triplificação, que consiste na descrição dos dados através de triplas RDF, seguindo a sintaxe sujeito, predicado e objeto, conforme um modelo de dados RDF[4] adaptado de [Maia and Yagui 2017]. O formato de triplas RDF torna-se uma escolha interessante devido à facilidade em se agregar novas informações ao modelo de dados e, conforme a necessidade, realizar consultas mais elaboradas ou mesmo integrar dados de outros domínios ao modelo proposto. Ao final deste processo ETL é realizada a carga das triplas no banco de dados para posterior extração de índices altmétricos.

2.3. Módulo de análise do impacto acadêmico

Este módulo é responsável pela extração das métricas de produtividade e impacto acadêmico, com base na RSC construída a partir do módulo descrito em 2.1. Assim como o módulo da seção 2.1, este módulo corresponde a uma implementação do método de análise e ranqueamento de pesquisadores descrito em [Maia et al. 2018]. O algoritmo de análise e ranqueamento do REALM permite a análise de RSC em três níveis:

(i) Global - mapeia a rede como um todo, o que permite comparar o comportamento de publicação e colaboração entre pesquisadores de diferentes áreas. Utiliza como parâmetros: número de pesquisadores da rede, número de publicações da rede, número de componentes da rede (sub-redes de nós conectados), somatório de publicações considerando cada pesquisador individualmente, somatório de pesquisadores considerando cada publicação individualmente, média de publicações por pesquisador, média de pesquisadores por publicação e média de colaboração (Grau Médio).

(ii) Local - mapeia as sub-redes existentes, o que permite identificar *clusters* de pesquisadores importantes. Utiliza como parâmetros: número de nós/elementos (NE), Grau Médio, diâmetro e densidade da sub-rede.

(iii) Individual – mapeia pesquisadores influentes a partir do número de publicações (NP) e de sua centralidade na rede, baseada no *Degree* (Grau), *Betweenness* (Intermediação), *Closeness* (Proximidade), e Pagerank. As métricas de centralidade foram escolhidas pela definição de 'prestígio' detalhada em Wasserman e Faust (1994). O prestígio de grau está associado à quantidade de vínculos diretos de um elemento na rede. Quanto mais vínculos o pesquisador tiver na RSC, maior seu prestígio de grau. O prestígio de proximidade considera como mais "centrais" aqueles que possuem uma distância média menor em relação a todos os outros da rede. Pesquisadores que colaboram com elementos mais centrais na RSC possuem proximidade maior. O prestígio de intermediação atribui maior prestígio aos elementos que são pontes, conectando diferentes grupos de pesquisa. Além disso, atribuímos maior status aos elementos mais referenciados na RSC, utilizando a métrica PageRank [Brin and Page 1998]. A visualização da RSC e o cálculo das métricas de centralidade foram adaptados da biblioteca *open source* Cystoscape.js[5].

[4]https://luisfmpm.github.io/realm/datamodel

[5]http://js.cytoscape.org/

Algorithm 1: Algoritmo de análise e ranqueamento do REALM.

```
   Data: array multidimensional sub-rede, NP, NE
   Result: array multidimensional sub-rede_ranqueada
 1 sub-rede_ranqueada ← [];
 2 foreach sub-rede as i do
 3 |   sub-rede[i]['deg'] ← degree(sub-rede[i]['no']) ; sub-rede[i]['bet'] ← betweenness(sub-rede[i]['no']) ;
 4 |   sub-rede[i]['clo'] ← closeness(sub-rede[i]['no']) ; sub-rede[i]['pag'] ← pagerank(sub-rede[i]['no']) ;
 5 end
 6 foreach sub-rede as i do
 7 |   sub-rede[i]['deg_pos'] ← posicao(sub-rede[i]['deg']) ; sub-rede[i]['bet_pos'] ← posicao(sub-rede[i]['bet']) ;
 8 |   sub-rede[i]['clo_pos'] ← posicao(sub-rede[i]['clo']) ; sub-rede[i]['pag_pos'] ← posicao(sub-rede[i]['pag']) ;
 9 end
10 foreach sub-rede as i do
11 |   sub-rede[i]['score'] ← (sub-rede[i]['deg_pos'] + sub-rede[i]['bet_pos'] + sub-rede[i]['clo_pos']) ;
12 end
13 sub-rede_ranqueada ← array_orderby(sub-rede, 'score', SORT_ASC, 'np', SORT_DESC) ; /* Ordena a
      sub-rede de forma crescente pelo score e usa como critério de desempate o maior NP    */
14 foreach sub-rede_ranqueada as i ⇒ var do
15 |   if (sub-rede_ranqueada[var]['np'] < NP) then
16 |   |   unset(sub-rede_ranqueada[i]); /* Remove o Pesquisador do ranking                  */
17 |   end
18 end
19 sub-rede_ranqueada ← array_slice(sub-rede_ranqueada, 0, NE);/* Limita o ranking pelo NE informado */
```

O algoritmo de análise e ranqueamento do REALM[6] (Algoritmo 1) também ordena os pesquisadores em suas respectivas sub-redes (caso haja grafos desconectados) utilizando como parâmetros o NP e as métricas de centralidade, sendo esses parâmetros configuráveis. Exemplo: ordenando somente os 100 primeiros colocados (NE=100) nas quatro métricas de centralidade e que possuam cinco ou mais publicações (NP >=5).

2.4. Módulo de análise do impacto social

Este módulo é responsável pela extração das métricas de impacto social, com base em consultas SPARQL[7] realizadas na base de triplas a partir da interface do sistema. Ele corresponde a uma implementação do método de análise da repercussão social de um pesquisador descrito em [Maia and Oliveira 2017]. A implementação consiste em três grupos de consultas SPARQL que são executadas no banco de triplas Apache Jena Fuseki[8] por meio de requisições HTTP intermediadas pela biblioteca *open source* EasyRDF[9]. Essas consultas são necessárias para a extração de índices altmétricos que permitem medir: (i) o alcance das pesquisas em veículos de comunicação primários (sites de notícias) e secundários (exemplo: blogs e fóruns científicos) (Consulta 1); (ii) sua penetração na população, através da disseminação em RSO como Facebook e Google+ (Consulta 2); e (iii) sua visibilidade a nível global, identificando o país de origem da publicação (Consulta 3). Para isso as consultas utilizam como parâmetros a quantidade de menções a um pesquisador em publicações, a quantidade de menções em publicações compartilhadas em RSO e a quantidade de menções por país, conforme mostrado na Tabela 1.

Conforme as consultas vão sendo realizadas as métricas de impacto social são extraídas e gravadas no banco de dados. A partir disso torna-se possível também indicar em que categoria de reputação um pesquisador se encontra, correlacionando o impacto acadêmico versus impacto social, sendo quatro categorias possíveis:

[6]O pseudocódigo do Algoritmo 1 refere-se ao trecho onde ocorre a análise individual da RSC.

[7]https://www.w3.org/TR/rdf-sparql-query/

[8]https://jena.apache.org/documentation/serving_data/

[9]http://www.easyrdf.org/docs

Tabela 1. Consultas (simplificadas) para extração dos indicadores altmétricos

Prefixos das consultas SPARQL com base no modelo de dados RDF

```
PREFIX ebucore: <https://www.ebu.ch/metadata/ontologies/ebucore/index.html#>
PREFIX schema: <http://schema.org/>
PREFIX realm: <https://luisfmpm.github.io/realm/datamodel#>
PREFIX dbo: <http://dbpedia.org/ontology/>
```

Consulta 1

```
SELECT DISTINCT ?noticia ?texto ?CountFB
?CountGPlus
WHERE {?noticia a ebucore:NewsItem.
       ?noticia schema:text ?texto.
       ?noticia realm:facebookCount ?CountFB.
       ?noticia realm:gplusCount ?CountGPlus.
       FILTER (CONTAINS(LCASE(str(?texto)),
       LCASE("Nome")))}
```

Consulta 2

```
SELECT DISTINCT (COUNT(?noticia) as ?TotalNot)
(SUM(?CountFB) as ?TotalFB)
(SUM(?CountGPlus) as ?TotalGPlus)
WHERE {?noticia a ebucore:NewsItem.
       ?noticia schema:text ?texto.
       ?noticia realm:facebookCount ?CountFB.
       ?noticia realm:gplusCount ?CountGPlus.
       FILTER (CONTAINS(LCASE(str(?texto)),
       LCASE("Nome")))}
```

Consulta 3

```
SELECT DISTINCT (COUNT(?noticia) as ?n)
?pais
WHERE {?noticia a ebucore:NewsItem.
       ?noticia schema:text ?texto.
       ?noticia dbo:country ?pais.
       FILTER (CONTAINS(LCASE(str
       (?texto)),LCASE("Nome")))}
GROUP BY (?pais)
ORDER BY DESC(?n)
```

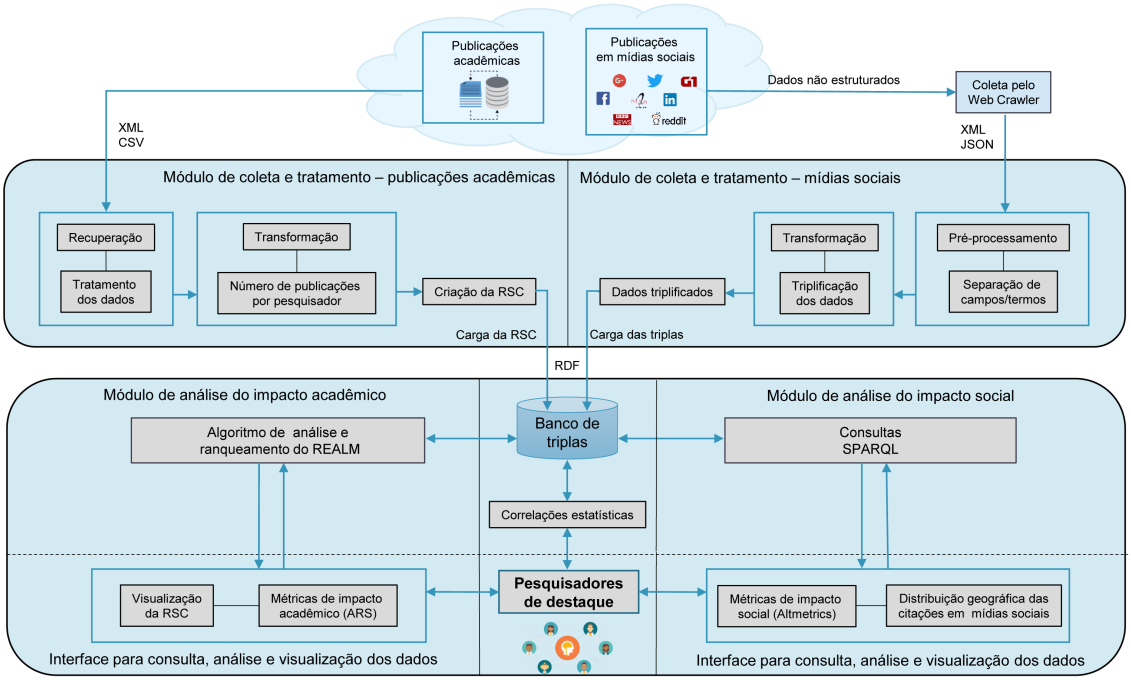


Figura 1. O *framework* computacional REALM

(i) Alto impacto acadêmico e social - Pesquisadores de destaque no cenário. São nomes de grande influência em sua área de atuação, pertencendo a redes de colaboração científica com fortes referências geopolítica/institucional e com forte presença on-line.

(ii) Alto impacto acadêmico - Geralmente são pesquisadores que integram núcleos de pesquisa bem definidos, porém com pouca presença on-line.

(iii) Alto impacto social – São pesquisadores que não possuem uma rede de colaboração bem definida, mas que frequentemente preferem outros meios de compartilhar seus resultados, como *fasttracks* e blogs científicos, o que torna sua divulgação mais prática e célere, sobretudo em mídias sociais.

(iv) Baixo impacto acadêmico e social - Pesquisadores de pouca importância no cenário e pouca ou nenhuma presença on-line.

Além da extração e visualização dos índices altmétricos, a principal contribuição deste módulo é a identificação dos nomes de destaque no cenário. A partir da interface um usuário do sistema pode configurar parâmetros de mapeamento, realizar consultas, acessar métricas gerais da RSC, dos clusters mapeados e dos pesquisadores ranqueados,

visualizar o grafo de colaboração da RSC, visualizar o mapa de citações dos pesquisadores e observar os nomes de destaque no cenário, além de fazer downloads desses dados. A Figura 1 ilustra o *framework* REALM.

3. Estudo de caso Piloto: ZIKV

Como dito na seção 1, a proposta do *framework* é servir como um novo método para entender a representatividade e reconhecimento dos pesquisadores perante a sociedade, a partir de *altmetrics*. Deste modo o *framework* foi testado no cenário do ZIKV, um cenário de emergência que forneceu grande riqueza de dados devido aos surtos ocorridos recentemente. Neste estudo, utilizamos o *framework* para tentar responder as seguintes perguntas: “Quem são os pesquisadores mais influentes academicamente?"e "Quem são os pesquisadores com maior inserção na população, possuindo um alto impacto social?”. Findadas nossas análises[10], nosso método foi submetido à prova de conceito, onde especialistas da área verificaram se os pesquisadores mais influentes identificados através do *framework* são, de fato, os nomes de maior reputação no cenário. O estudo divide-se em cinco etapas que serão explicadas a seguir.

Coleta de publicações sobre a Zika na base de dados indexada PubMed e construção da RSC – Nesta etapa utilizamos o mecanismo de consultas[11] do PubMed para recuperar publicações acerca do tema. A partir da *string*[12] “Zika” aplicada nos filtros ‘título’, ‘*abstract*’ e ‘texto’ da publicação, recuperamos os dados de 1.932 publicações retroativas a 21/12/2016, nos formatos XML e CSV. A partir desses dados foi possível construir a RSC da temática Zika, conforme as operações descritas em 2.1, para sua posterior leitura, visualização e extração de métricas de impacto acadêmico.

Coleta e triplificação de publicações sobre a Zika em mídias sociais - Nesta etapa o web *crawler* foi configurado para coletar publicações sobre a Zika (*query* “Zika”) em sites de notícias, blogs e fóruns de discussão, além de compartilhamentos em RSO como Facebook e Google+, em mais de 100 idiomas e durante o período de 28 de outubro a 28 de dezembro de 2016 (aproximadamente 62 dias de coleta). Neste processo foram coletados, triplificados (conforme descrito em 2.2) e armazenados os dados de 71.898 publicações sobre a Zika para compor uma base de dados temática.

Análise da reputação na Rede Social Científica – Esta etapa visa responder a primeira pergunta: “Quem são os pesquisadores mais influentes academicamente?". Para responder esta pergunta a RSC construída foi analisada em três níveis diferentes.

No primeiro nível foi realizada uma análise global, onde a rede foi verificada como um todo. Neste ponto, foram identificados 6.834 pesquisadores na RSC da doença Zika, onde pesquisadores do tema publicam em média 1,47 artigo, os artigos possuem uma média de 5,39 pesquisadores e a média de colaboração entre eles é de 6,12.

No segundo nível foi realizada uma análise local ou de grupos, onde as sub-redes que se formaram foram verificadas para identificarmos os grupos de pesquisadores mais importantes. Neste nível de análise foram identificados os três *clusters* de pesquisadores

[10]Por razões de escopo neste artigo não mostraremos o mapa da distribuição geográfica das citações.

[11]https://www.ncbi.nlm.nih.gov/pubmed/advanced

[12]((zika[Title/Abstract]) AND zika[Text Word]) AND ("1500/01/01"[Date - Publication] : "2016/12/21"[Date - Publication])

Tabela 2. Categorias de cores baseadas no número de publicações

Categoria	Condição	Categoria	Condição
Vermelha	If NP >= 5 OR NP < 8	Azul	If NP >= 10 OR NP < 15
Roxa	If NP >= 8 OR NP < 10	Verde	If NP >=15

mais importantes, que neste estudo serão referidos como sub-rede 1 (208 nós), sub-rede 2 (133 nós) e sub-rede 3 (96 nós).

No terceiro nível foi realizada uma análise individual, onde foram aplicadas as métricas de centralidade para identificação dos pesquisadores mais influentes dentro das três sub-redes. Para a análise da reputação de um pesquisador no cenário científico, foram utilizados os seguintes parâmetros: número de publicações e centralidade (*Betweenness*, *Closeness*, *Degree* e Pagerank) de cada pesquisador.

Com relação ao número de publicações é importante frisar que este critério deve ser levado em conta nas análises, pois a quantificação da produtividade, a despeito de críticas ao fato, também é um fator essencial para determinarmos se um pesquisador está conduzindo avanços em seu campo de atuação ou em focos específicos (para este estudo de caso, a doença Zika). Deste modo, o mapeamento foi configurado de modo a considerar somente pesquisadores com cinco ou mais publicações, descartando pesquisadores com baixa produção bibliográfica e reduzindo o escopo da próxima análise. Além disso, como forma de facilitar na identificação dos pesquisadores mais produtivos e melhorar a visualização desses números, foram definidas quatro categorias de cores baseadas no NP de cada pesquisador, conforme os critérios definidos na Tabela 2.

Definidos esses critérios, os pesquisadores mais influentes foram ranqueados pelo algoritmo com base nas métricas de centralização de Freeman [Freeman 1978] (*Betweenness*, *Closeness* e *Degree*), a partir do somatório das posições, e individualmente na métrica Pagerank. A última métrica é utilizada como critério de comparação, já que indica se um pesquisador está relacionado com nós que são bastante referenciados na RSC. Estes dois *rankings* encontram-se disponíveis em: https://goo.gl/tAjftd e https://goo.gl/3nLG4c.

Análise da repercussão social de um pesquisador - Esta etapa está relacionada à resolução da segunda pergunta: "Quem são os pesquisadores com maior inserção na população, possuindo um alto impacto social?" Para isso foram realizados dois tipos distintos de consulta, extraindo os índices altmétricos a partir de: (i) veículos de comunicação, para verificar se estão noticiando os avanços e descobertas de um pesquisador em relação ao ZIKV - para isto, utilizamos o total de menções (em sites de notícias, fóruns e blogs) a um pesquisador. (ii) RSO, para verificar a disseminação e alcance das publicações na população – para isto, analisamos a propagação da publicação sobre um pesquisador e suas descobertas a partir de menções no Facebook e no Google+.

Essas duas consultas indicam se: (i) os veículos de comunicação mais importantes (sites de notícias) e os secundários (blogs e fóruns científicos, por exemplo) estão noticiando os avanços e descobertas de um pesquisador em relação à doença. (ii) se essas notícias estão se disseminando pelas RSO e alcançando um público mais amplo.

Todavia, o nome de citação do PubMed é insuficiente para identificar os registros de citações de um pesquisador. Diante disso, foram utilizadas grafias alternativas para auxiliar nas consultas. Por exemplo, para o pesquisador DIAMOND MS, foram identifi-

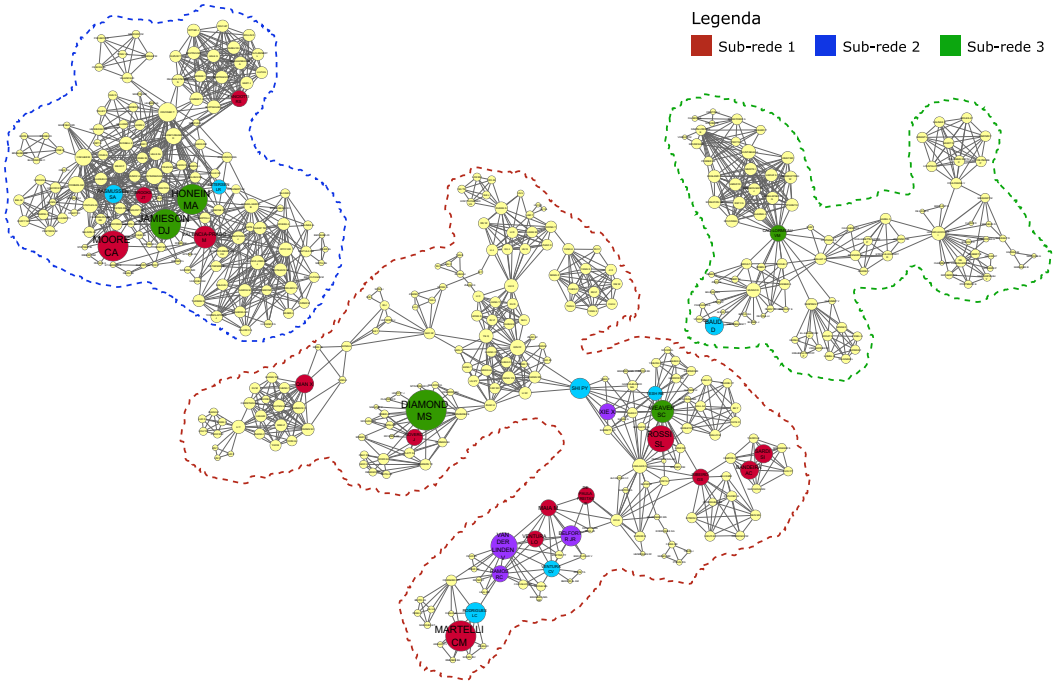


Figura 2. Pesquisadores de destaque no cenário da Zika

cadas três grafias diferentes na base de triplas, 'DIAMOND MS', 'Michael S. Diamond' e 'Michael Diamond', de modo que na primeira consulta as grafias foram testadas individualmente e fazendo uso da cláusula '*distinct*' para contabilizar somente uma referência por publicação. Com isso foram retornados, respectivamente, 6, 24 e 683 resultados para as três grafias identificadas (713 citações). Deste modo, a consulta foi realizada utilizando o operador || (OR) dentro da cláusula '*filter*' e em conjunto com '*distinct*' para retornar, em uma única consulta, os resultados das três grafias para o nome do pesquisador e garantir que não há resultados repetidos. Novamente foram retornados 713 resultados.

A partir disso, realizamos essas duas consultas para os pesquisadores identificados na etapa de análise da reputação na RSC, utilizando a cláusula '*Order by*' para ordená-los de acordo com o volume de menções em publicações (em sites de notícias, blogs e fóruns), e utilizando as menções em RSO (Facebook e Google+) como critério de desempate.

A partir da ordenação dos pesquisadores com base nesses critérios de classificação, foi criado um *ranking* altmétrico unificado das três sub-redes, que encontra-se disponível em https://goo.gl/8QyN6a.

Pesquisadores de destaque no cenário - Com os resultados obtidos também foi possível verificar a correlação entre impacto acadêmico e impacto social. A verificação ocorre a partir da correlação entre as variáveis *Betweenness*, *Closeness*, *Degree*, PageRank e citações em mídias sociais com resultados que variam numa escala entre 0 e 1, para o grupo das métricas de centralização versus as citações individuais na base de dados temática. Quanto mais próximo de 1, maior a correlação entre impacto acadêmico e impacto social. Deste modo, identificamos os 30 pesquisadores de destaque no cenário da Zika (disponíveis em: https://goo.gl/SMzx2b). A Figura 2 ilustra a RSC em Zika, onde estão presentes os 30 nomes de destaque no cenário, conforme as categorias de cores da Tabela 2 e o tamanho do nó variando conforme a importância do pesquisador.

4. Prova de conceito

Para avaliar o *framework*, foi realizada uma prova de conceito. Usando o *framework*, foram criados dois *rankings*, contendo: i) os 20 pesquisadores com maior influência científica e ii) os 20 pesquisadores com maior influência social.

As duas listas, ordenadas de maneira decrescente em relação às respectivas influências, foram apresentadas a três pesquisadores especialistas no domínio, participantes da Rede Zika de Ciências Sociais (chefiada pela Fiocruz) e da ZIKAlliance (consórcio internacional). Para não serem influenciados, foram apresentados a eles apenas as listagens. Nenhuma explicação sobre o *framework* foi dada. Os pesquisadores avaliaram separadamente os dois *rankings* e concordaram integralmente com as ordenações apresentadas.

Um dos pesquisadores (participante do primeiro grupo), que estuda e documenta a evolução da doença no Brasil e no mundo, justificou cada uma das posições dos *rankings*. Tal explicação foi feita baseando-se no seu conhecimento tácito e informações sobre a evolução científica da doença. Este pesquisador mostrou as influências de cada pesquisador (no contexto científico ou social) e demonstrou as diferenças entre suas relevâncias.

5. Trabalhos relacionados

Estudos empíricos no campo da Altmetria podem se basear em diferentes grupos de plataformas que permitem a extração de diferentes grupos de métricas de impacto acadêmico e social. Isso é natural se pensarmos que, fundamentalmente, as métricas alternativas fazem uso de índices que relacionam a quantidade de menções a cientistas e/ou suas pesquisas em diferentes plataformas.

Entre essas plataformas, podemos citar RSO como Facebook e Google+, blogs científicos (ou blogs em geral) e gerenciadores de referências bibliográficas como o Mendeley e CiteULike. Esta forma de quantificar a ciência está amparada por diversos estudos empíricos que visam demonstrar de maneira efetiva o impacto das pesquisas acadêmicas na sociedade em geral. Esta perspectiva é abordada nos trabalhos de: (i) Bornmann (2015), com a utilização de três tipos de plataformas, sendo a primeira a RSO Twitter, a segunda os gerenciadores de referências bibliográficas Mendeley e CiteULike, e a terceira blogs científicos; (ii) Hassan e Gillani (2016), que propuseram um estudo altmétrico baseado em diversas plataformas, como, Google Scholar, Twitter, Mendeley, Facebook, Google+, CiteULike, blogs e *Wiki*; (iii) Kwak e Lee (2014), que utilizaram o Twitter; (iv) Mohammadi et al.(2015) na plataforma Mendeley e (v) Hoffman et al. (2014) que utilizaram o Researchgate.

Neste sentido outro trabalho que merece destaque no campo da Altmetria é o Altmetric.com[13], sendo atualmente a ferramenta mais popular no que se refere a este tipo de medição. O Altmetric.com tem o propósito de rastrear e analisar a atividade on-line no que diz respeito à literatura acadêmica fornecendo feedback de dados de aproximadamente 5 milhões de *papers*. Seus serviços incluem a extensão Altmetric Bookmarklet[14], que quando instalada no navegador oferece métricas a nível de artigo, bastando para isso navegar até a página onde o artigo se encontra e clicar o botão "Altmetric it!" nos favoritos. O Altmetric Bookmarklet, no entanto, funciona apenas em artigos de páginas do PubMed, arXiv ou que possuam DOI.

[13]https://www.altmetric.com/

[14]https://www.altmetric.com/products/free-tools/bookmarklet/

Embora existam diversas ferramentas altmétricas disponíveis, as mais populares, como o Altmetric.com e o Altmetric Bookmarklet exigem o pagamento de taxas para sua utilização ou não satisfazem as necessidades de pesquisadores e instituições acadêmicas. Este é um cenário que incentiva o desenvolvimento de novas ferramentas altmétricas para atender às demandas mais específicas. Neste sentido, o *framework* apresentado se destaca dos demais trabalhos por suprir essa demanda através de uma infraestrutura de aplicações que permitem analisar a reputação de pesquisadores e pesquisas através de diferentes grupos de métricas, de maneira conjunta e com maior precisão do que outros métodos que utilizam somente um tipo de medição. Outra vantagem é a possibilidade de comparar a evolução de áreas a partir de aspectos temporais e macro das RSC construídas.

6. Conclusão

Neste trabalho apresentamos o framework REALM, que permitiu a extração conjunta de diferentes tipos métricas para avaliar a reputação de pesquisadores no cenário da Zika. A extração de três grupos de métricas a partir de uma única ferramenta representa um avanço no que tange a medição dos impactos da ciência, visto que métricas de produtividade, influência acadêmica e impacto social, sozinhas, podem não ser suficientes para evidenciar isso. O alto número de compartilhamentos em alguns casos indica que a pesquisa teve grande repercussão nas RSO (penetração na população). Estes casos referem-se a publicações que reportam progressos em estudos relacionados ao ZIKV e outras descobertas científicas importantes, como novos tratamentos e a proximidade de uma cura. Notamos que essas publicações remetem a estudos desenvolvidos por cientistas de alto prestígio acadêmico e social, sendo integrantes de núcleos de pesquisa bem definidos e de forte referência geopolítica e institucional, conforme corroborado na prova de conceito.

Além do próprio *framework*, este estudo fornece uma contribuição para o cenário da pesquisa em Zika, visto que até o momento não há mapeamentos de RSC acerca da doença a nível micro/individual, ou seja, analisando em profundidade as interações científicas sobre o tema. O *framework* desenvolvido para investigar essas questões se baseia em um conjunto de abordagens sistemáticas que, com a extração de três tipos de medição distintas e sua aplicação combinada, permite avaliar melhor os impactos de pesquisadores, sua produtividade e influência na comunidade acadêmica e na sociedade.

Outra característica do REALM é o uso de conceitos da web semântica para fusão e registro de dados, permitindo consultas mais direcionadas ao conteúdo pretendido. Além disso, diferente dos bancos relacionais, essa abordagem possibilita a inserção de novas categorias conforme a necessidade de ampliar o modelo e reorganizar as informações, permitindo flexibilidade dos dados. Por exemplo, agregando triplas de publicações sobre novas doenças não é necessário alterar a estrutura dos dados já registrados no banco.

Como trabalhos futuros, pretende-se estender o modelo do grafo RDF de modo a analisar impactos de pesquisas em outros cenários como a Dengue e Chikungunya e com maior volume de dados. Também é pretendido investigar o engajamento com o público, ou seja, quantas pessoas e que tipo de audiência lê e publica sobre esses temas.

Referências

Bornmann, L. (2014). Validity of altmetrics data for measuring societal impact: A study using data from Altmetric and F1000prime. *J. Informetr.*, 8(4):935–950.

Bornmann, L. (2015). Alternative Metrics in Scientometrics: A Meta-analysis of Research into Three Altmetrics. *Scientometrics*, 103(3):1123–1144.

Brin, S. and Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. and ISDN systems*, 30(1):107–117.

Freeman, L. C. (1978). Centrality in social networks conceptual clarification. *Soc. Networks*, 1(3):215–239.

Harmon, A. (2016). Handful of Biologists Went Rogue and Published Directly to Internet. *The NYT*.

Hassan, S.-U. and Gillani, U. A. (2016). Altmetrics of "altmetrics"using Google Scholar, Twitter, Mendeley, Facebook, Google-plus, CiteULike, Blogs and Wiki. *arXiv:1603.07992 [cs]*.

Hoffmann, C. P., Lutz, C., and Meckel, M. (2014). Impact Factor 2.0: Applying Social Network Analysis to Scientific Impact Assessment. In *Proceedings of the 47th Int. Conf. on Syst. Sciences*, pages 1576–1585. IEEE.

Kwak, H. and Lee, J. G. (2014). Has Much Potential but Biased: Exploring the Scholarly Landscape in Twitter. In *Proceedings of the 23rd Int. Conf. on World Wide Web*, pages 563–564, New York, NY, USA. ACM.

Maia, L. F. M. P., Lenzi, M., Rabello, E. T., and Oliveira, J. (2018). Colaborações científicas em Zika: Identificação dos principais grupos e pesquisadores através da análise de redes sociais. *Cad. de Saúde Pública*.

Maia, L. F. M. P. and Oliveira, J. (2017). Investigation of research impacts on the Zika virus. An approach focusing on social network analysis and altmetrics. In *Proceedings of the 23rd Brazillian Symp.on Multimedia and the Web*, Gramado.

Maia, L. F. M. P. and Yagui, M. M. M. (2017). Triplificação de dados de notícias sobre a Zika. In *Proceedings of the XIII Brazilian Symp. on Information Systems*, Lavras.

McNeil Jr, D. G. (2016). Zika Data From the Lab, and Right to the Web. *The NYT*.

Mohammadi, E., Thelwall, M., Haustein, S., and Larivière, V. (2015). Who reads research articles? An altmetrics analysis of Mendeley user categories. *J Assn Inf Sci Tec*, 66(9):1832–1846.

Oliveira, C. S. and Vasconcelos, P. F. C. (2016). Microcephaly and Zika virus. *J. Pediatr.*, 92(2):103–105.

Priem, J. (2013). Scholarship: Beyond the paper. *Nature*, 495(7442):437–440.

Priem, J., Groth, P., and Taraborelli, D. (2012). The Altmetrics Collection. *PLoS One*.

Priem, J. and Hemminger, B. H. (2010). Scientometrics 2.0: New metrics of scholarly impact on the social Web. *First Monday*, 15(7).

Wasserman, S. and Faust, K. (1994). *Social network analysis: Methods and applications*, volume 8. Cambridge university press.

Zanluca, C., Melo, V. C. A., Mosimann, A. L. P., et al. (2015). First report of autochthonous transmission of Zika virus in Brazil. *Mem. Inst. Oswaldo Cruz*, 110(4):569–572.

Meta-alinhamento de ontologias utilizando a abordagem presa-predador

Nicolas Ferranti[1], **Stênio Sã Rosário Furtado Soares**[1], **Jairo F. De Souza**[1]

[1] Departamento de Ciência da Computação – Universidade Federal de Juiz de Fora (UFJF)
36.036-900 – Juiz de Fora, MG – Brazil

{nicolas1, ssoares}@ice.ufjf.br, jairo.souza@ufjf.edu.br

Abstract. *Every year, several new ontology matchers are proposed in the literature, each one using a different heuristic, which implies in different performances according to the characteristics of the ontologies. An ontology meta-matcher consists of an algorithm that combines several approaches in order to obtain better results in different scenarios. To achieve this goal, it is necessary to define a criterion for the use of matchers. We presented in this work an ontology meta-matcher that combines several ontology matchers making use of the evolutionary meta-heuristic prey-predator as a means of parameterization of the same.*

Resumo. *Todo ano, diversos novos alinhadores de ontologias são propostos na literatura, cada um utilizando uma heurística diferente, o que implica em desempenhos distintos de acordo com as características das ontologias. Um meta-alinhador consiste de um algoritmo que combina diversas abordagens a fim de obter melhores resultados em diferentes cenários. Para atingir esse objetivo, é necessária a definição de um critério para melhor uso de alinhadores. Neste trabalho, é apresentado um meta-alinhador de ontologias que combina vários alinhadores através da meta-heurística evolutiva presa-predador como meio de parametrização das mesmas.*

1. Introdução

Ontologias são construídas por pessoas com diversos níveis de especialização e visão de domínio. Logo, conceitos que podem descrever o mesmo tipo de objeto podem se encontrar representados de formas distintas, tanto na sintaxe dos termos quanto na estrutura de relações, gerando um problema de heterogeneidade na semântica dos dados. Para solucionar problemas de heterogeneidade, é preciso uma forma de especificar, sem ambiguidade, os vocabulários subjacentes aos sistemas de informação [Farinelli and Almeida 2014].

O alinhamento de ontologias é uma etapa fundamental em aplicações tradicionais da área de banco de dados que por natureza lidam com estruturas heterogêneas, em alguns sistemas o processo é considerado uma etapa prévia fundamental para o uso do sistema como o processo de *message mapping* [Hai et al. 2007], em outros, como os sistemas Web que fazem uso de bases de Dados Ligados, a operação de alinhamento é necessária em tempo de execução [Souza et al. 2014].

O problema a ser resolvido consiste de definir relações entre conceitos das ontologias envolvidas, compatibilizando as estruturas de forma a representar a união dos conjuntos de dados em um novo modelo. O alinhamento de ontologias, como é denominado, é

um problema complexo e suas características possibilitam que seja abordado por diversas técnicas computacionais. Devido à alta heterogeneidade das ontologias, não existe uma técnica que se sobressaia dentre as outras em todos os aspectos [Xue and Tang 2017]. Logo, abordagens de meta-alinhamento podem ser utilizadas neste cenário. Um meta-alinhador combina diversas técnicas de alinhamento, a fim de explorar vários aspectos da heterogeneidade para evitar que o desempenho do alinhamento seja restrito a alguma característica das ontologias. A literatura sugere que os melhores resultados encontrados no meta-alinhamento de ontologias estão associados ao uso de algoritmos evolucionários devido à sua capacidade de adaptação e, consequentemente, adequação do uso de cada técnica [Souza 2012, Shi and Eberhart 1998, Xue and Tang 2017].

O uso de meta-heurísticas em meta-alinhadores é justificada pelo tamanho do espaço de busca do problema, que desencoraja o uso de abordagens exaustivas devido ao tempo de processamento demandado. Além disso, a literatura mostra que abordagens populacionais apresentam-se como boas alternativas na solução do problema.Embora o uso de Algoritmos Genéticos seja mais frequente entre as abordagens, outras abordagens bio-inspiradas constituem um campo promissor a ser exploradas na solução o problema. Na última década, muitas meta-heurísticas foram propostas e ainda são pouco exploradas na área de banco de dados [Sorensen et al. 2017].

Dentre as abordagens populacionais bio-inspiradas, o algoritmo presa-predador (PPA) possui características que são adequadas para o problema de meta-alinhamento, uma vez que permite que regiões promissoras do espaço de soluções sejam exploradas por um conjunto de agentes (soluções), pressionados a fugirem de regiões pouco atraentes em termos de valor da função objetivo, ao mesmo tempo que permite a exploração de novas regiões ao se atribuir a esses agentes um comportamento pseudoaleatório na definição do seu deslocamento.

O objetivo deste trabalho é tratar o problema de meta-alinhamento de ontologias através de uma adaptação do algoritmo presa-predador (PPA) e mostrar sua aplicabilidade para esse problema. Para analisar o comportamento da solução, foi utilizado o *benchmark* fornecido pela *OAEI (Ontology Alignment Evaluation Initiative*). Observou-se que o PPA é eficiente e eficaz ao parametrizar as técnicas de alinhamento de forma a obter uma solução que é próxima da ótima em tempo polinomial. Assim, o trabalho mostra que o PPA pode ser melhor explorado no cenário de alinhamento de ontologias.

2. Técnicas e Meta-alinhamento

O problema de alinhamento de ontologias pode ser tratado de diversas formas. Esta seção traz uma revisão dos métodos de alinhamento e meta-alinhamento presentes na literatura. Segundo [Otero-Cerdeira et al. 2015], técnicas de alinhamento de ontologias podem ser classificadas em níveis, seguindo duas lógicas de interpretação: como as técnicas lidam com a entrada fornecida, englobando alinhadores em nível dos elementos e em nível estrutural e a interpretação baseada no tipo da entrada fornecida, contemplando técnicas baseadas em conteúdo e em contexto. A especificação de cada tipo é apresentada a seguir:

- Alinhadores em nível dos elementos: técnicas que obtêm as correspondências considerando as entidades nas ontologias isoladamente, ignorando que são partes da estrutura da ontologia.

- Alinhadores a nível estrutural: técnicas que obtêm as correspondências analisando como as entidades se encaixam dentro da estrutura das ontologias.
- Baseadas em conteúdo: técnicas com foco na informação interna proveniente das ontologias que serão alinhadas.
- Baseadas em contexto: consideram para a correspondência as informações externas que podem surgir de relações entre ontologias ou outros recursos externos (contexto).

Com base no sistema de classificação presente em [Otero-Cerdeira et al. 2015], é possível agrupar as técnicas presentes na literatura por categoria, tornando mais fácil a comparação entre as mesmas. Em [Akbari et al. 2009], os autores apresentam uma medida de similaridade baseada na distância de Levenshtein para a comparação de *strings* aplicada em alinhamento de ontologias. Técnicas de comparação de *strings* fazem uso, por exemplo, do rótulo e da descrição das entidades da ontologia para determinar relações. Existem várias métricas para cálculo de distância que podem ser usadas nesses métodos, como Jaccard, n-gram, Levenshtein, TFIDF, euclidiana etc. Em [Joslyn et al. 2009] foram aplicadas técnicas da Teoria dos Grafos ao problema. Essas técnicas consideram as ontologias a serem alinhadas como grafos rotulados, ou até mesmo árvores, caindo no problema de isomorfismo entre dois grafos. Observa-se, entretanto, que não existe uma técnica que se sobressaia às demais de forma genérica e seja eficaz em todos os casos [Souza 2012]. Em alguns casos, a comparação de strings identifica facilmente a equivalência entre nomes semelhantes, enquanto a análise do grafo pode indicar que, devido às relações encontradas, duas classes das ontologias são equivalentes. Logo, é interessante estudar o uso dessas técnicas em conjunto para melhorar a acurácia de um alinhamento.

Com o uso de técnicas de alinhamento híbridas, o Lily [Wang and Wang 2016] é capaz de resolver alguns problemas relacionados a ontologias heterogêneas. Os resultados alcançados pela abordagem no *benchmark* da OAEI na campanha de 2016 foram superiores ou iguais a todos os outros alinhadores em relação à Medida-F [Achichi et al. 2016]. O Lily constrói um subgrafo semântico na tentativa de eliminar a interpretação heterogênea dos elementos das ontologias. Todo o cálculo de similaridade é realizado sobre esse subgrafo. A similaridade é computada por meio de técnicas de similaridade entre strings e similaridade estrutural. Ao final, as similaridades computadas são combinadas utilizando pesos experimentalmente definidos. Os pesos são fundamentais para definir o nível de confiança para uma dada abordagem e, nesse caso, são atribuídos estaticamente, prejudicando o desempenho da solução em ontologias cuja experimentação não foi aplicada. Uma alternativa viável é o desenvolvimento de meios para que o algoritmo possa se adaptar dinamicamente a uma dada entrada e atribuir automaticamente os pesos mais adequados para aquela ontologia. Abordagens que calibram o peso de técnicas de alinhamento em tempo de execução são denominadas meta-alinhadoras de ontologias.

O termo meta-alinhamento de ontologias [Euzenat et al. 2007] descreve sistemas que parametrizam automaticamente um conjunto de funções de alinhamento de ontologias. [Martinez-Gil and Aldana-Montes 2012] define um conjunto de características comuns no que tange aos meta-alinhadores de ontologias: (1) Não é necessário que o processo de meta-alinhamento seja realizado em tempo de execução. As funções de alinhamento podem ser computadas em *background* e aplicadas em tempo de execução uma vez que o processo executado por elas é determinista e as relações não mudam de uma execução para a outra; (2) O processo de meta-alinhamento deve ser automático, logo,

deve ser possível que seja implementado por alguma ferramenta de alinhamento; (3) O processo deve se comportar como um especialista, caso a melhor função de alinhamento não seja conhecida, o processo deve ser capaz de experimentar pesos e combinações a fim de retornar a função mais próxima possível da melhor função de alinhamento; e (4) Uma estratégia de meta-alinhamento é avaliada com a função de alinhamento retornada.

O meta-alinhamento lida com a integração de alinhadores heterogêneos, visando encontrar os melhores parâmetros que possam afetar os resultados do alinhamento. O problema é modelado como um problema de otimização e as abordagens mais importantes empregam heurísticas em conjunto com algoritmos evolutivos, gulosos ou baseados em conjuntos de regras [Souza et al. 2014]. Como a literatura não apresenta o uso da meta-heurística presa-predador para a calibração de funções de alinhamento, e algoritmos evolucionários são frequentemente empregados para esse fim, este trabalho experimentou o uso do presa-predador para a calibragem das funções de alinhamento.

3. Trabalhos Relacionados

O problema de meta-alinhamento de ontologias é um problema relativamente recente e que ainda possui várias características a serem exploradas, ainda que as propostas conhecidas tenham apresentado bons resultados. Nesta seção são apresentados trabalhos relacionados que fazem uso de meta-heurísticas para tratar de alinhamento de ontologias, destacando suas principais características e contribuições.

O uso de meta-heurística tem sido explorado para resolver o problema de meta-alinhamento, como em [Souza 2012, Xue and Tang 2017, Bock and Hettenhausen 2012]. No GNoSIS+ [Souza 2012], é utilizado algoritmo genéticos para parametrizar um conjunto preestabelecido de alinhadores. O aprendizado do algoritmo é baseado em um grupo de alinhamentos de referência definidos na entrada por um engenheiro de ontologias. A premissa é que alguns relacionamentos podem ser facilmente apontados, então o AG calibra as funções do sistema baseado na referência a fim de prepará-lo para uma situação real de aplicação. É interessante destacar a representação do problema pelo GNoSIS+. Considerando $\Xi = \{F_1, F_2, ..., F_n\}$ um conjunto de funções de alinhamento, cada cromossomo possui *n* genes($|\Xi| = n$) e cada gene representa um valor real $w \in [0, 1]$ que representa o peso a ser aplicado sobre cada função. O objetivo é minimizar a diferença entre o valor encontrado e o valor definido pelo engenheiro de ontologias para um relacionamento em específico. [Xue and Tang 2017] também emprega um algoritmo evolucionário com a mesma representação de indivíduo, entretanto, a função objetivo passa a ser maximizar o valor da média harmônica da Medida-F. A Medida-F é uma medida que leva em conta as taxas de precisão e cobertura entre os mapeamentos obtidos pelo algoritmo com os que eram esperados. A função objetivo de cada trabalho guia os respectivos algoritmos para caminhos diferentes. Para a abordagem de [Xue and Tang 2017], é necessário avaliar cada item do resultado obtido a cada iteração com a base de referência, acarretando em um custo computacional maior do que apenas comparar o resultado obtido com o valor de confiança definido pelo engenheiro, como é feito em [Souza 2012].

O MapPSO [Bock and Hettenhausen 2012] é uma solução que emprega a técnica de enxame de partículas para lidar com o problema de meta-alinhamento. O enxame de partículas é uma técnica com inspiração natural baseada no comportamento social de indivíduos, como, por exemplo, a revoada de pássaros para encontrar um local com ali-

mento suficiente [Shi and Eberhart 1998]. A abordagem busca apenas relações do tipo equivalência (1:1) e utiliza um alinhador predefinido que implementa uma função de distância. A função de distância define um nível de similaridade para um dado par de conceitos. É importante notar que o MapPSO não calibra um conjunto de funções alinhadoras, pois utiliza apenas uma. Entretanto pode ser considerado uma abordagem de meta-alinhamento por buscar um alinhamento ótimo fazendo uso de alinhadores predefinidos. A representação do indivíduo difere dos trabalhos anteriores. Nesta abordagem, cada solução é representada como um alinhamento candidato. Suponha que $\vec{X_p}$ represente um alinhamento de duas ontologias constituído de *k* = 5 correspondências (c). A partícula é representada por $\vec{X_p} = \left\{c_{(p,1)}, c_{(p,2)}, c_{(p,3)}, c_{(p,4)}, c_{(p,5)}\right\}$ onde cada $c_{(p,i)}$ indica um valor confiança para o relacionamento (p, i). A função objetivo do MapPSO busca encontrar a maior quantidade de alinhamentos possíveis, podendo prejudicar o desempenho em ontologias onde, por natureza, a taxa de correspondências é baixa.

A literatura mostra que abordagens evolutivas apresentam bons resultados quando aplicadas no alinhamento de ontologias, como é o caso do GOAL [Martinez-Gil and Aldana-Montes 2011] que obteve 97% de medida-f sobre o *benchmark* da OAEI, o que permite fomentar que o uso da meta-heurística presa-predador tem potencial para construir resultados efetivos para o problema. A definição da representação do indivíduo impacta na forma como o esforço da abordagem pode ser reproduzido. Uma vez que a representação seja baseada no conjunto de pesos, os parâmetros encontrados podem ser armazenados e recuperados sem muito esforço, enquanto que a representação baseada no conjunto de alinhamentos candidatos requer que todo o processo seja executado novamente. Logo, os pesos das abordagens de conjuntos de pesos tem contribuição melhor para a construção de um meta-alinhador mais genérico.

4. Solução Proposta

Para tratar o problema, é aplicada a meta-heurística presa-predador baseada na interação entre animais. A primeira versão da meta-heurística foi introduzida inicialmente por [Laumanns et al. 1998], onde as presas não se movem naturalmente, estando sujeitas ao movimento dos predadores para que então possam responder de forma a se adaptar no espaço de busca, melhorando a qualidade da solução. [Tilahun and Ong 2015] apresentam uma abordagem também baseada na interação presa-predador entre animais, entretanto, o comportamento e a forma como os indivíduos interagem entre si se difere dos outros trabalhos. O trabalho de [Tilahun and Ong 2015] apresenta o comportamento da meta-heurística presa-predador que é adaptada para neste trabalho.

Considere S um conjunto de correspondências conhecidas de equivalência. O conjunto S é formado por tuplas $(e_{1i}, e_{2i}, =, s_i)$, onde e_{1i} e e_{2i} são entidades de ontologias distintas, $=$ denota a relação do tipo equivalência e s_i é a similaridade conhecida, informada pelo engenheiro de ontologias, entre e_{1i} e e_{2i}. Seja f uma função de similaridade composta da soma ponderada de outras funções, ao aplicar a função f em e_{1i} e e_{2i}, espera-se encontrar o valor s_i, ou seja, $f(e_{1i}, e_{2i}) = s_i$. Como exemplo, considere o conjunto $S' = (e_{11}, e_{21}, =, 1), (e_{12}, e_{22}, =, 1), (e_{13}, e_{23}, =, 1)$, com todas as correspondências possuindo similaridade igual a 1. Considerando uma função $\bar{f}'(e_{11}, e_{21}) = g_1(e_{11}, e_{21})p_1 + g_2(e_{11}, e_{21})p_2 + g_3(e_{11}, e_{21})p_3$, onde g_i representa o valor de similaridade definido pela função i que são constantes do problema e p_i representa o peso atribuído à função i. Logo, para cada alinhamento conhecido fornecido na entrada,

é possível construir um sistema linear:

$$\begin{array}{rcl} \bar{f}'(e_{11}, e_{21}) & = & s_1 \therefore g_1(e_{11}, e_{21})p_1 + g_2(e_{11}, e_{21})p_2 + g_3(e_{11}, e_{21})p_3 = 1 \\ \bar{f}'(e_{12}, e_{22}) & = & s_2 \therefore g_1(e_{12}, e_{22})p_1 + g_2(e_{12}, e_{22})p_2 + g_3(e_{12}, e_{22})p_3 = 1 \\ \bar{f}'(e_{13}, e_{23}) & = & s_3 \therefore g_1(e_{13}, e_{23})p_1 + g_2(e_{13}, e_{23})p_2 + g_3(e_{13}, e_{23})p_3 = 1 \end{array} \quad (1)$$

O objetivo é encontrar os melhores valores de p_i de forma a minimizar a soma das diferenças entre o valor encontrado e o valor esperado. Como a meta-heurística presa-predador é populacional, a representação do indivíduo é baseada no conjunto de pesos p_i. Para modelar o problema, cada indivíduo recebe um conjunto de valores reais $p_i \in [0, 1]$ cuja alteração no valor impacta diretamente na confiança atribuída às funções associadas.

Um conjunto de soluções p_i viáveis é construído de forma aleatória, onde o somatório de todos pesos de cada solução não deve ultrapassar 1. Para cada solução x_i, é atribuído um valor de sobrevivência $SV(x_i)$, calculado à partir da função objetivo do problema. Seja $F(x)$ a função objetivo descrita como a soma das diferenças, ou seja, onde quanto menor o valor, melhor é a avaliação da solução, definimos:

$$SV(x_i) = 1/F(x_i) \quad (2)$$

Isso pode ser considerado como quão bem localizada está uma presa para fugir de um predador, no caso onde $F(x_i)$ é 0 ou próximo de 0, o que seria o resultado ótimo, $F(x_i)$ recebe uma constante da ordem de 1^{-10}. Após o valor de sobrevivência (SV) de cada membro da solução ser calculado, o membro com o menor SV será designado como um predador e o resto como presas. Uma vez que as presas e o predador são definidos, as presas precisam fugir do predador e tentar seguir as melhores presas em termos de valores de sobrevivência ou encontrar um bom esconderijo ao mesmo tempo. O que leva à definição de como se dará esse movimento. Ao tratar do movimento, é preciso definir duas questões: a direção e o tamanho do passo.

4.1. Cálculo da direção

Considerando que as presas precisam fugir ou tentar se esconder, é sorteado um número dentro de uma probabilidade fixa que define se a presa deve seguir as mais aptas ou procurar se esconder na vizinhança. Caso uma presa x_i escolha seguir as demais, tomando $\{x_1, x_2, ..., x_p\}$ como o conjunto de presas com valor de sobrevivência maior que x_i, o cálculo da nova direção é dado pela Equação 3, onde r_{ij} representa a distância entre as duas presas e τ um valor escolhido para ponderar o peso do valor de sobrevivência. Se a probabilidade de seguir não for alcançada, uma direção aleatória y_r é construída e então avaliada na presa x_i calculando a distância para o predador.

$$y_i = \sum_{j=1}^{p} e^{SV(x_j)^\tau - r_{ij}}(x_j - x_i) \quad (3)$$

$$d_1 = \|x_{predador} - (x_i + y_i)\| \quad (4)$$

$$d_2 = \|x_{predador} - (x_i - y_i)\| \quad (5)$$

Se $d_1 < d_2$ então tomar a direção $-y_r$ faz com que a presa x_i fique mais distante do predador, caso contrário é utilizada y_r. Após os cálculos de direção, fugindo ou seguindo, é necessário calcular quanto o indivíduo vai caminhar na direção encontrada.

4.2. Cálculo do tamanho do passo

O tamanho do passo define o quão longe a presa vai caminhar na direção escolhida. Ressalta-se que a natureza do problema de meta-alinhamento é contínua, tornando inviável a exploração de todo o espaço de busca. Logo, a definição do passo é importante para que não se perca uma boa solução no meio do caminho. Uma vez que uma presa longe do predador não correrá tão rápido quanto uma perto, o passo é dado por:

$$\lambda_i = \frac{\lambda_{MAX}\varepsilon_1}{e^{\beta\left|SV(x_i)-SV(x_{predador})\right|^{\omega}}} \tag{6}$$

Onde, λ_{MAX} representa o maior tamanho do passo, ε_1 um número escolhido randomicamente de forma uniforme no intervalo $[0, 1]$ e as constantes β e ω são definidas previamente antes da execução do algoritmo. Foi acrescentado um valor de granularidade (G) na equação do passo que contribui para controle do salto, generalizando a equação de movimento das presas temos, para a direção y_i adequada à escolha de seguir ou fugir:

$$x_i \leftarrow x_i + G\lambda_i(\frac{y_i}{\|y_i\|}) \tag{7}$$

O predador sempre se movimentará na direção da presa com pior SV, com um certo nível de aleatoriedade, como descreve a Equação 3.8:

$$x_{predador} \leftarrow x_{predador} + \lambda_{MAX}(\varepsilon_5)(\frac{y_r}{\|y_r\|}) + \lambda_{MIN}(\varepsilon_6)(\frac{x_i^{'} - x_{predador}}{\left\|x_i^{'} - x_{predador}\right\|}) \tag{8}$$

Onde λ_{MIN} e λ_{MAX} são constantes definidas previamente representando o passo mínimo e máximo respectivamente, ε_5 e ε_6 são valores reais aleatórios no intervalo [0,1], y_r uma direção gerada randomicamente e $x_i^{'}$ representa a posição da pior presa.

4.3. Intensificação da solução

Caso a presa em avaliação seja a de melhor SV em toda a população, não ocorre caminhamento. Segundo [Tilahun and Ong 2015], é aconselhável que nesta presa seja executado um processo de intensificação da solução a cada iteração. Na nossa solução, é utilizada uma busca local que percorre todos os pesos de uma solução criando duas novas soluções para cada peso visitado. O processo se dá pela soma e subtração do valor de granularidade G em cada peso do indivíduo, ou seja, se x_i é um indivíduo com conjunto de pesos $(g_1, ..., g_n)$, ao iterar sobre o primeiro peso são criadas duas novas soluções $(g_1+G, ..., g_n)$ e $(g_1 - G, ..., g_n)$. O processo é executado para cada peso e, ao final, o melhor aprimorante é escolhido para substituir o antigo indivíduo se sua aptidão for superior. Com a definição das funções de movimento e de intensificação, os passos do algoritmo podem ser especificados como:

1. Definir os parâmetros e gerar um conjunto de soluções viáveis

2. Calcular o valor de sobrevivência para cada presa e definir a melhor presa, o predador e as presas restantes
3. Fazer com que as presas e o predador se movimentem
4. Se o critério de parada for atendido, terminar a execução, senão, voltar ao passo 2

No intuito de diversificar a população criada pelo algoritmo, foi definido um número $\kappa = 10$ que representa a quantidade de vezes que o processo deve se repetir, executando todos os passos desde a criação da população até a busca local ao final, caso seja determinado.

4.4. Integração do calibrador

O trabalho desenvolvido faz uso da ferramenta de alinhamento apresentada em [Souza 2012]. A ferramenta possui arquitetura distribuída, incorpora um conjunto de funções de similaridade sintática que podem ser selecionadas para uso no meta-alinhamento e permite que o módulo calibrador de parâmetros seja substituído por outras abordagens, desde que o padrão das mensagens seja mantido e respeitando o fluxo de execução. O fluxo de execução se inicia com os arquivos de entrada. Considerando que a abordagem é supervisionada, cada teste precisa informar o conjunto de pré-alinhamentos de treinamento, juntamente com o par de ontologias, o alinhamento de referência completo para avaliação final e por último, o conjunto de funções que serão utilizadas para treinar o algoritmo. As funções de alinhamento são pré-cadastradas no sistema e fazem uso de métricas distintas para avaliar aspectos diversificados das entidades da ontologia. Algumas das métricas das funções presentes no conjunto teste são apresentadas a seguir:

- Similaridade das entidades com base na similaridade entre os comentários que descrevem as entidades
- Similaridade baseado na semelhança entre os termos da entidade
- Similaridade com base nos identificadores em comum que compõem um subgrafo das relações de cada entidade
- Similaridade das entidades com base nas instâncias de mesmo identificador
- Similaridade das entidades com base na semelhança entre os termos que identificam propriedades de tipo de dado e de objeto

Em seguida, o sistema linear é criado e as estruturas que avaliarão os pesos encontrados. É chamado o calibrador com a abordagem presa-predador e, assim, o meta-alinhador computa os alinhamentos candidatos utilizando os pesos encontrados e retornando as equivalências mais relevantes. Para escolher quais alinhamentos são mais relevantes, foi adotado um método que computa as funções e os pesos para cada par de entidades candidatas. O método ordena pelos maiores graus de similaridade encontrados e seleciona sempre o maior par como relacionamento escolhido, removendo os escolhidos do restante da lista. A Figura 1 representa a sequência dos dados dentro do sistema.

5. Avaliação

Para avaliação da abordagem, foi utilizado o *benchmark* da OAEI, a qual é utilizada para avaliar abordagens de alinhamento, mas que pode também ser utilizado por meta-alinhadores. Das bases disponíveis no benchmark, foi escolhida a base dentro do domínio de referências bibliográficas. Nesta base, a ontologia de referência é descrita sobre a linguagem OWL-DL e serializada em RDF/XML. Esta base é adequada para analisar meta-alinhadores porque possui um conjunto de testes que representam alterações sistemáticas

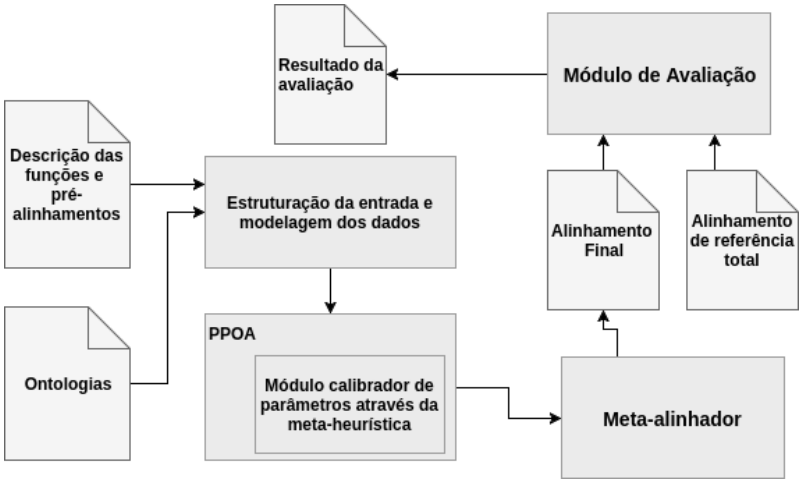


Figura 1. Fluxograma de execução do algoritmo

na ontologia de referência, de forma a analisar qualidades e defeitos de alinhadores de ontologias. No contexto de meta-alinhamento, os testes podem indicar o quão bem uma abordagem de meta-alinhamento se adapta para diferentes cenários. Ao todo, são 56 casos de testes, divididos em 3 categorias, a qual chamamos de 1xx, 2xx e 3xx. Os testes 1xx são testes no nível de linguagem de descrição, os teste 2xx são testes de alterações sistemáticas nos elementos da ontologia (rótulos, relações, propriedades, instâncias e hierarquia), e os teste 3xx analisam alinhamentos com ontologias externas. É importante destacar que a faixa de testes 3xx representa o cenário mais próximo do real, pois objetiva alinhar a ontologia de referência com ontologias reais sobre referências bibliográficas.

Para a execução desse experimento, foram desenvolvidas três versões da abordagem. A primeira versão, denotada por PP Simples, não realiza a busca local. Assim, apenas as operações de movimento foram executadas, fazendo com que a melhor presa permanecesse imóvel até que seja ultrapassada por uma melhor solução. A segunda versão, denotada por PP Simples + BLI, faz com que a melhor solução seja submetida à busca local, avaliando a vizinhança e seguindo para o melhor aprimorante de forma iterativa, enquanto um vizinho melhor pudesse ser encontrado. Já a terceira versão, denotada por PPA, reproduz o comportamento do algoritmo com o movimento da população de acordo com o que foi apresentado e uma busca local simples na melhor presa, fazendo apenas um movimento na direção do melhor vizinho uma única vez a cada rodada. Para os experimentos, as constantes foram ajustadas como: probabilidade de seguir em 50%, $G = 0.005$, $\lambda_{MIN} = 1$, $\lambda_{MAX} = 20$, $\tau = 0.09$, $\beta = 1$, $\omega = 1$. Onde os valores de β e ω são recomendados por [Tilahun and Ong 2015] e os demais definidos empiricamente.

O sistema foi avaliado através das métricas de Precisão, Cobertura e *Medida-F*. A Tabela 1 apresenta a média de cada métrica para os três modelos dentro das faixas de teste do *benchmark*. O melhor resultado observado foi o do terceiro modelo, que contempla o movimento dos indivíduos no espaço de solução e a busca local executada na melhor presa que retorna o melhor vizinho imediato a essa presa, caso o mesmo exista.

Como o terceiro modelo apresentou resultado médio melhor que os outros, foi realizada uma análise mais aprofundada neste modelo. Uma vez que o algoritmo possui fatores randômicos, a execução foi repetida a fim de avaliar a estabilidade do modelo. Os

Tabela 1. Resultados dos testes nos três modelos

		Valores Médios dentro da faixa		
	Faixa de instâncias	Precisão	Cobertura	*Medida-F*
PP Simples	101-104	1,000	1,000	1,000
	201-247	0,920	0,920	0,920
	301-304	0,763	0,736	0,749
	Média Total:	**0,913**	**0,915**	**0,914**
PP Simples+BLI	101-104	1,000	1,000	1,000
	201-247	0,939	0,946	0,942
	301-304	0,823	0,790	0,806
	Média Total:	**0,936**	**0,938**	**0,937**
PPA	101-104	1,000	1,000	1,000
	201-247	0,946	0,952	0,949
	301-304	0,820	0,786	0,802
	Média Total:	**0,941**	**0,943**	**0,942**

resultados médios para um conjunto de execuções são apontados na Tabela 2.

Tabela 2. Resultados médios do conjunto de execuções no modelo PPA

	Média	Desvio Padrão
Precisão	0,90678125	0,0562465884
Cobertura	0,90844791	0,0560783869
Medida-F	0,90703632	0,0558457086

Ao comparar os resultados do PP Simples com o PP Simples+BLI, é possível ver que a busca local iterativa tem impacto positivo na qualidade das soluções, como apontam as medidas das faixas 2xx e 3xx, aprimorando a melhor solução encontrada pelo PP Simples. O resultado esperado foi encontrado, onde a aplicação da meta-heurística da forma como foi definida (PPA) superou as demais, o meta-alinhador utilizando o presa-predador como calibrador de pesos apresentou resultados promissores nos parâmetros empregados, tais resultados tendem a melhorar com a customização das funções de movimento, uma proposta de trabalhos futuros.

No que tange ao cenário de aplicação, o PPA pode ser empregado onde é necessário atribuir alinhamentos candidatos a ontologias com poucas referências e retornar uma primeira versão de alinhamento para os engenheiros. Outro cenário, onde espera-se que o desempenho seja melhor, é aquele em que se deseja refazer um alinhamento já existente dado que houve alguma alteração em uma das ontologias. A expectativa de melhores resultados se baseia no fato de que o conjunto de alinhamentos de referência de entrada é maior, provendo uma capacidade descritiva maior para o algoritmo. Da maneira como foi desenvolvido, o PPA permite que o usuário possa controlar o esforço da abordagem, através dos parâmetros, isso significa que a solução pode ser produzida de acordo com a necessidade do utilizador, por consequência a qualidade da solução também sofre impacto. Logo, com a configuração adequada é possível que o PPA possa ser utilizado em tempo de execução, o que não foi um problema nos testes realizados, dado o tamanho das ontologias. Ontologias com maior volume de informação, de larga escala, requerem

um tempo maior tanto no calibramento quanto no meta-alinhamento em si. Como este trabalho executa as duas tarefas em tempo de execução, a performance do sistema tende a cair, podendo inviabilizar sua aplicação nesse cenário. A OAEI possui conjuntos de testes que lidam com o alinhamento de ontologias em larga escala.

6. Considerações Finais

Este trabalho apresentou uma abordagem para meta-alinhamento de ontologias que faz uso da meta-heurística presa-predador. A natureza do problema em conjunto com a forma como foi modelado permite que, quando uma boa solução for encontrada, os pesos associados a essa solução possam ser persistidos e utilizados para reproduzir o experimento em ontologias que atendam aos critérios estabelecidos pelo teste. Uma vez que os relacionamentos entre as entidades das ontologias estão estabelecidos, não há necessidade de reprocessar o algoritmo para encontrar uma nova solução, como ocorre, por exemplo, em problemas de roteamento de veículos, onde novas soluções devem ser geradas com frequência pois as restrições variam ao longo do tempo. Logo, encontrar uma solução perto da ótima dentro de um conjunto de execuções em tempo polinomial é suficiente, pois os parâmetros encontrados podem ser aplicados em qualquer momento.

O método de alinhamento utilizado, que seleciona sempre o par de entidades com maior similaridade, tende a ser substituído em trabalhos futuros, uma vez que um método que explore as similaridades encontradas e busque o conjunto de alinhamentos que maximize o somatório das similaridades envolvidas pode contribuir com uma melhor estabilidade do modelo, considerando que a similaridade correta pode não ser a de maior valor encontrado mas está situada entre as melhores.

Como limitações do trabalho, podemos destacar que o *benchmark* da OAEI é criado de forma sistemática com base em dados sintéticos, o que dá margem para que possam ocorrer variações de desempenho e acurácia para ontologias do mundo real. Ainda assim, o *benchmark* é difundido na literatura, sendo a principal referência para testes de alinhadores de ontologias. O desempenho do algoritmo desenvolvido está atrelado às configurações definidas pelo usuário. Logo, é importante realizar um estudo sobre os valores ideais desses parâmetros. Ainda, a abordagem proposta está baseada em uma única função objetivo. Porém, objetivos conflitantes poderiam ser definidos, por exemplo, adicionar como objetivo a maximização do número de correspondências totais. Assim, abordagens multi-objetivos, conhecidas de outras áreas da computação, poderiam ser exploradas e adaptadas para essa abordagem.

Referências

Achichi, M., Cheatham, M., Dragisic, Z., Euzenat, J., Faria, D., Ferrara, A., Flouris, G., Fundulaki, I., Harrow, I., Ivanova, V., et al. (2016). Results of the ontology alignment evaluation initiative 2016. In *OM: Ontology Matching*, pages 73–129. No commercial editor.

Akbari, I., Fathian, M., and Badie, K. (2009). An improved mlma+ and its application in ontology matching. In *Innovative technologies in intelligent systems and industrial applications, 2009. CITISIA 2009*, pages 56–60. IEEE.

Bock, J. and Hettenhausen, J. (2012). Discrete particle swarm optimisation for ontology alignment. *Information Sciences*, 192:152–173.

Euzenat, J., Shvaiko, P., et al. (2007). *Ontology matching*, volume 18. Springer.

Farinelli, F. and Almeida, M. (2014). Interoperabilidade semântica em sistemas de informação de saúde por meio de ontologias formais e informais: um estudo da norma openehr. *XVII Encontro Nacional de Pesquisa em Ciência da Informação*, 17(1).

Hai, D. et al. (2007). *Schema matching and mapping-based data integration: Architecture, approaches and evaluation*. VDM Verlag.

Joslyn, C. A., Paulson, P., and White, A. (2009). Measuring the structural preservation of semantic hierarchy alignments. In *Proceedings of the 4th International Conference on Ontology Matching-Volume 551*, pages 61–72. CEUR-WS. org.

Laumanns, M., Rudolph, G., and Schwefel, H.-P. (1998). A spatial predator-prey approach to multi-objective optimization: A preliminary study. In *Parallel Problem Solving from Nature—PPSN V*, pages 241–249. Springer.

Martinez-Gil, J. and Aldana-Montes, J. F. (2011). Evaluation of two heuristic approaches to solve the ontology meta-matching problem. *Knowledge and Information Systems*, 26(2):225–247.

Martinez-Gil, J. and Aldana-Montes, J. F. (2012). An overview of current ontology meta-matching solutions. *The Knowledge Engineering Review*, 27(4):393–412.

Otero-Cerdeira, L., Rodríguez-Martínez, F. J., and Gómez-Rodríguez, A. (2015). Ontology matching: A literature review. *Expert Systems with Applications*, 42(2):949–971.

Shi, Y. and Eberhart, R. (1998). A modified particle swarm optimizer. In *IEEE International Conference on Evolutionary Computation*, pages 69–73.

Sorensen, K., Sevaux, M., and Glover, F. (2017). A history of metaheuristics. *Handbook of Heuristics*.

Souza, J. F. (2012). *Uma abordagem heurística uni-objetivo para calibragem em meta-alinhadores de ontologias*. PhD thesis, Pontifícia Universidade Católica do Rio de Janeiro.

Souza, J. F., Siqueira, S. W. M., Melo, R. N., and de Lucena, C. J. P. (2014). Analise de abordagens populacionais para meta-alinhamento de ontologias. In *iSys-Revista Brasileira de Sistemas de Informação*, pages 75–97.

Tilahun, S. L. and Ong, H. C. (2015). Prey-predator algorithm: A new metaheuristic algorithm for optimization problems. *International Journal of Information Technology & Decision Making*, 14(06):1331–1352.

Wang, P. and Wang, W. (2016). Lily results for oaei 2016. In *OM@ ISWC*, pages 178–184.

Xue, X. and Tang, Z. (2017). An evolutionary algorithm based ontology matching system. *Journal of Information Hiding and Multimedia Signal Processing*.

Melhorias no Processo de Blocagem para Resolução de Entidades Baseadas na Relevância dos Termos

Laís Soares Caldeira[1], **Anderson Almeida Ferreira**[1]

[1]Departamento de Computação – Universidade Federal de Ouro Preto (UFOP)
Ouro Preto, MG – Brasil

laissoarescaldeira@hotmail.com, anderson.ferreira@ufop.edu.br

Abstract. *Entity Resolution is a task commonly faced in data integration process. Due to quadratic number of comparisons to decide those instances belonging to the same entity, we need another way for performing such comparisons. In order to mitigate such a problem, techniques of blocking and block processing have been applied aiming the efficiency. In this work, we propose options to choose terms in the blocking step based on their relevance to the dataset in the phases of blocking and processing of blocks. We assess our proposal comparing it against relevant works available in the literature. The results show that our proposal decrease the run time by half, increasing the efficiency.*

Resumo. *Resolução de Entidades é uma tarefa comumente enfrentada no processo de integração de dados. Por necessitar de um número de comparações de ordem quadrática, torna-se inviável aplica-lá em grandes conjuntos de dados. Técnicas de blocagem e de processamentos de blocos têm sido propostas, visando amenizar esse problema. Neste trabalho, é proposta uma forma de escolher termos para serem usados na etapa de blocagem e no processamento de blocos, com base em sua relevância na coleção de dados. A proposta é avaliada comparando-a com trabalhos relevantes publicados na literatura. Os resultados mostram que a proposta deste trabalho reduz o tempo de processamento pela metade e melhora a qualidade dos blocos gerados.*

1. Introdução

A Web é um universo em crescimento e vem sendo dominada por conteúdo semi-estruturado e não estruturado. Além do aumento do montante de dados na *Web*, há uma grande diversidade entre as estruturas desses dados, dando origem a um gigantesco volume de dados heterogêneos. Tal questão representa um dos maiores desafios para busca na Web, levando a utilização de técnicas de integração de dados para melhorar os resultados retornados por essas buscas [Madhavan et al. 2007]. No processo de integração de dados, informações de diversas fontes devem ser comparadas e combinadas para que os usuários possam acessá-las e manipulá-las de forma unificada [Halevy et al. 2006].

Uma questão central do processo de integração de dados em larga escala é a Resolução de Entidades (ER - *Entity Resolution*), ou seja, a tarefa de identificar diferentes instâncias que pertencem a mesma entidade do mundo real [Christen 2012]. Uma entidade pode ser uma pessoa, uma empresa ou qualquer outro objeto com significado bem definido. Tipicamente a ER compara cada instância de uma coleção de dados com todas as outras, ou seja, a quantidade de comparações é de ordem quadrática com relação a quantidade de instâncias, tornando-se impraticável em grandes coleções de dados.

Para a ER se tornar escalável, normalmente são utilizadas técnicas de blocagem (conhecidas como técnicas de *blocking* - em inglês) [Christen 2012]. As técnicas de blocagem podem ser usadas para aprimorar o tempo de processamento em ER dividindo as instâncias em blocos, para comparar apenas as instâncias dentro do mesmo bloco. Com isso, o ganho em usar blocos está na redução de comparações entre as instâncias.

Os blocos podem ser reprocessados por técnicas de processamentos de blocos. Tais técnicas tentam reduzir comparações redundantes (instâncias comparadas várias vezes) e supérfluas (entre instâncias pertencentes a entidades distintas). Meta-blocagem (*meta-blocking* - em inglês) é uma técnica de reestruturação de blocos que descarta drasticamente comparações redundantes e supérfluas, por meio da transformação do conjunto de blocos em um grafo, onde os vértices correspondem às instâncias e as arestas conectam vértices que representam instâncias que coocorrem em um bloco, e objetiva manter apenas as arestas mais promissoras de correspondência [Papadakis et al. 2014]. Melhorias significativas na eficiência são encontradas com a aplicação de meta-blocagem. Porém, há muito o que se investigar para que a precisão das técnicas relacionadas ao processo de manipulação de blocos seja melhorada e que o tempo de processamento seja reduzido.

O trabalho tem como objetivo principal melhorar a eficiência (tempo de construção dos blocos), sem diminuir (podendo melhorar) a eficácia, em termos de correspondência entre instâncias encontradas, de técnicas de blocagem usadas em processos de ER, evitando comparações desnecessárias entre instâncias de uma coleção. Assim, o foco do trabalho é o processo de blocagem e não a tarefa inteira de Resolução de Entidades.

Contribuições: Inspirado nas abordagens baseadas em *tokens* (ou seja, termos) para formação dos blocos, a hipótese principal deste trabalho é que, características dos termos presentes nas instâncias da coleção de dados podem ser úteis para gerar blocos mais indicados para a ER. Assim, a originalidade do trabalho está no fato de avaliar e usar características específicas dos termos para alcançar melhorias na eficiência e na eficácia do processo de blocagem e seu processamento. Diversas características extraídas de termos foram analisadas e experimentadas, sendo que as características que obtiveram melhores resultados são apresentadas neste trabalho. Assim, este trabalho apresenta melhorias no processo de blocagem por meio do PBBRT (Processo de Blocagem Baseado na Relevância de Termos), que se divide em dois passos. Primeiramente, foi desenvolvida uma forma de escolher os termos a serem usados para blocagem por meio da entropia dos termos na coleção de dados. Com os blocos gerados no passo anterior, uma técnica de processamento de blocos que se baseia em meta-blocagem é adaptada utilizando a frequência dos termos na coleção de dados. Em conjunto, os dois passos são usados para produzir conjuntos de blocos de alta qualidade. O PBBRT foi avaliado em 3 coleções de dados semi-estruturados do mundo real (podendo ser aplicado a dados estruturados), mostrando resultados satisfatórios em relação a precisão e ao tempo de processamento, comparados aos da técnica de meta-blocagem proposta em Papadakis et al. [2016].

O restante deste trabalho está estruturado da seguinte forma: A Seção 2 descreve alguns conceitos importantes para o trabalho. A Seção 3 apresenta os trabalhos relacionados. A Seção 4 descreve a proposta deste trabalho. A Seção 5 descreve os experimentos e analisa os resultados. Finalmente, a Seção 6 apresenta conclusões trabalhos futuros.

2. Fundamentação Teórica

Nesta seção, são discutidos alguns conceitos fundamentais para o trabalho.

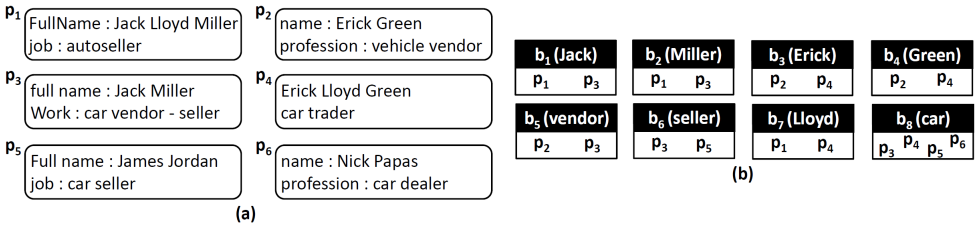


Figura 1. (a) Um conjunto de instâncias, e (b) os blocos resultantes da aplicação da técnica *Token Blocking*. Figura extraída de Papadakis et al. (2016).

2.1. Resolução de Entidades

No contexto de Resolução de Entidades (ER), uma instância p de um conjunto de dados P se refere a uma entidade e é descrita por meio de uma lista de atributos. Duas instâncias, p_i e p_j, são consideradas *duplicatas* quando p_i e p_j descrevem ou se referem à mesma entidade ($p_i \equiv p_j$). O objetivo da resolução de entidades é identificar todas as instâncias que são duplicatas no conjunto P, onde $D(P)$ representa esse conjunto de duplicatas e $|D(P)|$ a quantidade de duplicatas em $D(P)$ [Papadakis et al. 2016].

Há dois tipos de tarefas de ER: *Clean-Clean ER* e *Dirty ER*. A tarefa *Clean-Clean ER* recebe dois conjuntos sem duplicatas, mas que se sobrepõem, e identifica as instâncias que correspondem entre os conjuntos, enquanto a tarefa *Dirty ER* recebe como entrada um único conjunto com duplicatas e produz como saída um conjunto de grupos contendo cada um instâncias que são correspondentes [Papadakis et al. 2014].

2.2. Blocagem

Técnicas de blocagem aprimoram o tempo de processamento da ER. A maioria das técnicas de blocagem lidam com alto nível de heterogeneidade, tanto nos valores quanto nos nomes dos atributos. Isso é contornado normalmente ignorando as informações sobre o esquema e a semântica. Por exemplo, a técnica *Token Blocking* [Papadakis et al. 2013] é uma técnica de blocagem que lida com essa questão, colocando em um mesmo bloco instâncias que compartilham pelo menos um termo nos valores de seus atributos.

O exemplo da Figura 1 ilustra a técnica *Token Blocking*. A Figura 1 (a) contém as instâncias p_1, p_2, p_3, p_4, p_5 e p_6, em que p_1 corresponde a p_3 e p_2 corresponde a p_4, ou seja, pares que correspondem a mesma entidade (duplicatas). *Token Blocking* agrupa as instâncias nos blocos mostrados na Figura 1 (b). É possível notar que ambos os pares de duplicatas coocorrem em pelo menos um bloco, gerando um total de 13 comparações (par a par em cada bloco). A abordagem de força bruta, aquela que compara uma instância com todas as outras sem formar os blocos, teria 15 comparações.

No exemplo, os blocos b_1 e b_3 contêm uma comparação redundante e b_2 e b_4 também. Todos os outros blocos possuem comparações supérfluas entre instâncias não correspondentes, exceto para a comparação redundante p_3-p_5 em b_8, que se repete em b_6. No total, os blocos da Figura 1 (b) envolvem 2 comparações necessárias, 3 redundantes e 8 supérfluas, dentre as 13 comparações. Isso pode ser considerado uma proporção elevada. Neste trabalho, um conjunto de blocos será representado por B, com $|B|$ denotando seu tamanho (número de blocos) e $||B||$ sua cardinalidade (número total de comparações).

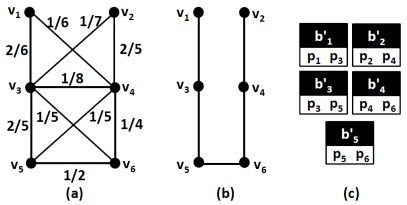


Figura 2. (a) Grafo de blocagem extraído dos blocos da Figura 1 (b), (b) um possível grafo de blocagem com arestas podadas, e (c) os novos blocos derivados. Figura extraída de Papadakis et al. (2016).

2.3. Meta-blocagem

Os blocos resultantes da blocagem podem ser reestruturados pela meta-blocagem [Papadakis et al. 2014]. Ela transforma um conjunto de blocos B em um grafo de blocagem GB, que contém um vértice para cada instância da coleção de dados remanescente da blocagem e uma aresta para cada par de instâncias de um bloco.

A Figura 2 (a) mostra o grafo para os blocos na Figura 1 (b) para uma estratégia de meta-blocagem. Cada aresta no grafo é ponderada com um peso análogo à probabilidade de que as instâncias ligadas pela aresta se referem a mesma entidade. Quanto maior o peso de uma aresta, mais provável é que as instâncias representadas nos vértices conectados sejam correspondentes. Como exemplo para a poda das arestas, pode-se definir como limiar a média dos pesos de todas as arestas do grafo. O grafo podado é mostrado na Figura 2 (b). O conjunto de blocos reestruturados B' é formado por meio da criação de um novo bloco para cada aresta mantida. Note que na Figura 2 (c) existem 5 blocos referentes às 5 arestas do grafo da Figura 2 (b). No entanto, o conjunto de blocos B' reduz as comparações de 13 para apenas 5, mantendo originalmente o número de possíveis duplicatas encontradas. Observe que a meta-blocagem tenta podar as arestas do grafo de blocagem deixando os vértices de instâncias correspondentes conectados.

A meta-blocagem descarta parte das arestas do grafo de blocagem utilizando um algoritmo de poda centrado nas arestas ou um algoritmo de poda centrado nos vértices do grafo. Em [Papadakis et al. 2014], os autores propuseram 4 opções de poda: *Cardinality Edge Pruning* (CEP), que ordena as arestas em ordem decrescente de peso e mantém somente as k primeiras, sendo $k = \lfloor \sum_{b \in B} |b|/2 \rfloor$ e $|b|$ o tamanho de cada bloco pertencente ao conjunto B; *Cardinality Node Pruning* (CNP), que mantém as *top-k* arestas da vizinhança de um vértice, sendo $k = \lfloor \sum_{b \in B} |b|/|P| - 1 \rfloor$ e $|P|$ a quantidade de instâncias no conjunto P; *Weighted Edge Pruning* (WEP), que descarta todas as arestas com peso menor que um limiar; e *Weighted Node Pruning* (WNP), que considera cada vértice do grafo e suas arestas adjacentes, podando as arestas que são inferiores a um limiar local.

O exemplo da Figura 2 utiliza um algoritmo de poda centrado nas arestas do grafo de blocagem. Um exemplo de poda centrada nos vértices do grafo é apresentado na Figura 3 (a). Para cada vértice na Figura 2 (a), foram mantidas as arestas incidentes que excedam o peso médio dos vértices vizinhos (vizinhança). Para maior clareza, as arestas mantidas são dirigidas, uma vez que podem ser mantidas na vizinhança de ambas as instâncias incidentes. Novamente, cada aresta mantida forma um novo bloco, obtendo o conjunto de blocos reestruturados B' da Figura 3 (b). Neste caso, o conjunto de blocos B' reduz as comparações de 13 para 9 em relação ao conjunto de blocos da Figura 1 (b).

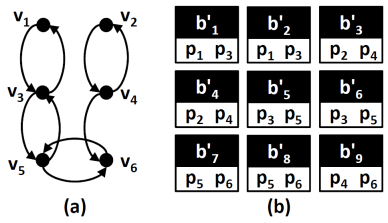


Figura 3. (a) Um possível grafo de blocagem com poda centrada em vértice, para o grafo da Figura 2 (a). (b) Os novos blocos derivados do grafo podado. Figura extraída de Papadakis et al. (2016).

3. Trabalhos Relacionados

Técnicas de blocagem e técnicas de processamento de blocos são frequentemente utilizadas em ER. A primeira técnica de blocagem encontrada na literatura é a *Standard Blocking* [Fellegi and Sunter 1969]. Nessa técnica, cada instância é representada por apenas uma chave de blocagem predefinida, agrupando as instâncias em blocos que compartilham exatamente a mesma chave. Desta forma, *Standard Blocking* é considerada uma técnica livre de redundância, pois produz blocos que não se sobrepõem. As técnicas de blocagem, em sua maioria, produzem blocos que se sobrepõe. Em Christen [2012], são apresentadas análises comparativas de uma grande parte das técnicas de blocagem. De acordo com [Papadakis et al. 2014], dependendo da interpretação de redundância, técnicas de blocagem podem ser classificadas em três categorias: técnicas de redundância positiva, técnicas de redundância negativa e técnicas de redundância neutra.

Técnicas de redundância positiva garantem que, se existem dois ou mais blocos com pares de instâncias, é provável que elas sejam correspondentes. Dentro desta categoria se encaixam as técnicas *Token Blocking* (descrita na Seção 2.2) e *Attribute Clustering* [Papadakis et al. 2013]. A técnica *Attribute Clustering* explora padrões nos nomes dos atributos na coleção de dados, e utiliza tais informações para construção dos blocos. As técnicas de redundância negativa garantem que as instâncias correferentes compartilham apenas um bloco, por exemplo, *Canopy Clustering* [McCallum et al. 2000] utiliza uma métrica de distância para colocar instâncias em blocos sobrepostos. As técnicas de redundância neutra produzem blocos sobrepostos, mas o número de blocos comuns entre duas instâncias é irrelevante para a probabilidade de correferência. *Sorted Neighborhood* [Hernández and Stolfo 1995] é um exemplo de técnica para essa categoria, que, por meio de uma janela deslizante de tamanho fixo, passa gradualmente sobre todas as instâncias da coleção de dados, criando os blocos dinamicamente.

As técnicas de processamento de blocos destinam-se a processar um conjunto de blocos já existentes, visando diminuir a quantidade de comparações entre as instâncias. São exemplos: *Iterative Blocking* [Whang et al. 2009] e *Comparison Propagation* [Papadakis et al. 2013]. *Iterative Blocking* distribui as duplicatas encontradas em um bloco a outros blocos que irão ser posteriormente processados, gerando resultados de novas instâncias coreferentes em outros blocos, tornando todo o processo iterativo. *Comparison Propagation* utiliza uma estrutura de tabela *hash* que possibilita descartar todas as comparações redundantes de um conjunto de blocos, mantendo as duplicatas encontradas. Meta-Blocagem, descrita na Seção 2.3, é considerada também uma técnica de processamento de blocos. As contribuições apresentadas em [Papadakis et al. 2016] para técnicas de meta-blocagem relacionadas a tarefa *Dirty ER* são comparadas a deste trabalho.

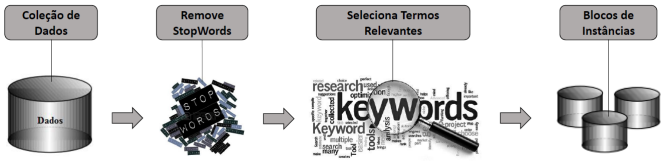


Figura 4. Etapas da construção de blocos do PBBRT

Simonini et al. [2016] propõem o BLAST. BLAST utiliza uma estratégia baseada em LSH (*Locality-Sensitive Hashing*) que coleta informações estatísticas diretamente dos dados. Com base nessas informações, os atributos são particionados de acordo com a semelhança de seus valores e, em seguida, é aplicada a técnica *Token Blocking* explorando os atributos de particionamento. Assim, apenas as instâncias cujos *tokens* pertencem a atributos na mesma partição serão comparadas. Posteriormente, usa-se essa informação na aplicação da técnica de meta-blocagem WNP, produzindo conjunto de blocos de alta qualidade, direcionados para a tarefa *Clean-Clean ER*.

A principal diferença da proposta deste trabalho para os demais está no fato de usar características de termos para a obtenção de blocos iniciais e a aplicação de características de termos no processamento dos blocos. Além disso, a proposta deste trabalho tem como foco a tarefa *Dirty ER*, diferentemente do trabalho de Simonini et al. [2016], que foca na tarefa *Clean-Clean ER*. Em [Papadakis et al. 2016], ambas as tarefas ER são abordadas.

4. PBBRT - Processo de Blocagem Baseado na Relevância dos Termos

Nesta seção, são apresentadas as melhorias no processo de blocagem por meio do PBBRT (Processo de Blocagem Baseado na Relevância dos Termos). O PBBRT é composto por duas técnicas que são utilizadas em conjunto: uma técnica de blocagem e uma técnica de processamentos de blocos. A seguir, as técnicas são retratadas em mais detalhes.

4.1. Blocagem do PBBRT

Para lidar com a complexidade do espaço de informações altamente heterogêneo, a técnica de blocagem do PBBRT se baseia na redundância positiva, onde cada instância pode ser colocada em vários blocos, de forma que quanto mais blocos duas instâncias estiverem, maior será a probabilidade das instâncias serem correspondentes. Isso é feito para reduzir o número de correspondências perdidas e é praticamente indispensável no contexto de dados heterogêneos. Para realizar a blocagem de um conjunto de instâncias, PBBRT verifica a relevância dos termos dessas instâncias e cria blocos onde os termos relevantes são as chaves. Assim, cada bloco contém as instâncias em que sua chave correspondente está presente nos valores dos atributos.

A Figura 4 ilustra as etapas da técnica de construção de blocos do PBBRT. A primeira etapa efetua a remoção de *stopwords*. *stopwords* são termos não representativos em uma coleção de dados. Geralmente esses termos são: preposições, artigos, advérbios, números, pronomes e pontuação [Wilbur and Sirotkin 1992].

Após a remoção de *stopwords*, é realizada a seleção de termos relevantes. Para identificar os termos relevantes foram analisadas e experimentadas diversas características relacionadas aos termos, tais como, entropia, quantidade de caracteres nos termos e frequência do termo na coleção. Dentre elas, a entropia mostrou melhores resultados e é a usada para blocagem neste trabalho. O conceito de entropia foi transformado numa

medida de quantidade de informação por [Shannon 2001]. Neste trabalho, considere entropia (H) calculada como H = - $\sum prob_i \times \log_2 prob_i$, sendo $prob_i$ a probabilidade do termo i estimada pela sua frequência na coleção de dados. O cálculo da probabilidade dos termos, $prob_i$, seria f_i dividido pela frequência total de todos os termos, $FreqTotal$. Portanto $prob_i = \frac{f_i}{FreqTotal}$. Dado que as probabilidades para todos os valores de frequência existentes foram estimadas, é realizado o somatório para encontrar H.

A ideia neste trabalho é fazer a blocagem com os termos com o maior ganho de informação, com o intuito de construir blocos representativos. Quanto maior o valor de H, maior é o ganho. Para saber a importância que uma determinado termo tem para uma coleção de dados, foram realizados cálculos de entropia desconsiderando os termos com uma determinada frequência em análise. Visto que cada cálculo de entropia desconsidera um valor de frequência dos termos, quanto menor for o resultado do cálculo, maior é a importância dos termos com a frequência excluída. Dessa forma, consideraram-se os termos excluídos quando a entropia fica menor que um limiar para serem chaves na geração de blocos. O limiar foi definido como a média de todos os valores de entropia encontrados.

Por fim, a criação dos blocos segue a técnica *Token Blocking*, descrita na Seção 2.2, considerando apenas os termos selecionados.

4.2. Processamento de Blocos do PBBRT

Os quatro esquemas de poda para a meta-blocagem, apresentados na Seção 2.3 (CEP, CNP, WEP e WNP), foram investigados em [Papadakis et al. 2016], onde mostrou-se que CNP e WNP geralmente são mais eficientes. Dado que o CNP supera WNP em termos de precisão, um algoritmo alternativo para o CNP, chamado *Reciprocal Node-centric Pruning*, abreviado como *Reciprocal* CNP, foi apresentado em [Papadakis et al. 2016] para aplicações ER com o objetivo de melhorar a eficiência do processo. Considerando que o presente trabalho tem foco em melhorar também a precisão, uma adaptação foi feita no *Reciprocal* CNP para receber como entrada o conjunto de blocos criados por meio da técnica de blocagem, apresentada na subseção anterior, visando atingir melhores resultados relacionados à eficiência. Na subseção a seguir é apresentado o funcionamento do *Reciprocal* CNP e na Subseção 4.2.2 é mostrada a adaptação realizada neste trabalho.

4.2.1. Reciprocal CNP

Na Subseção 2.3, foi descrita que a meta-blocagem transforma o conjunto de blocos resultante da técnica de blocagem em um grafo. No entanto, é muito custoso materializar o grafo de blocagem em memória, visto que, para grandes coleções de dados, o grafo pode conter milhares de vértices e arestas. Uma solução foi implementar o grafo implicitamente, integrando ao *Reciprocal* CNP o *Comparison Propagation* [Papadakis et al. 2013]. Nessa técnica, um índice de instâncias é construído. Esse índice constitui uma estrutura de tabela *hash*, cujas chaves são os identificadores das instâncias da coleção de dados remanescentes da blocagem e seus valores são listas com índices dos blocos que contêm as instâncias correspondentes.

Dessa forma, ao invés de iterar sobre todas as comparações nos blocos B de entrada, *Reciprocal* CNP itera sobre todas as instâncias, dado que o índice de instâncias foi criado. Para cada instância p_i (correspondente a um vértice no grafo), é identificada todas as outras instâncias que coocorrem com p_i nos blocos associados (vizinhança) e o

valor de frequência de cada um desses pares de instâncias é registrado em um vetor. Ao final, tem-se o número de blocos compartilhados por p_i e p_j e essa informação é utilizada para estimar o peso da aresta que liga as instâncias p_i e p_j. Dado que a vizinhança armazena cada par único de instâncias que coocorrem, a redundância pode ser eliminada. *Reciprocal* CNP trata as comparações redundantes como pares de instâncias com grandes chances de correspondência. Essas comparações correspondem a ligações recíprocas no grafo de blocagem. Por exemplo, as arestas $a_{1,3}$ e $a_{3,1}$ na Figura 3 (a) indicam que p_1 tem alta probabilidade de correspondência com p_3 e vice-versa, reforçando assim, a probabilidade de que as duas instâncias são duplicatas. Com base neste raciocínio, *Reciprocal* CNP mantém uma comparação para cada par de instâncias que são mutuamente ligadas no grafo de blocagem. Instâncias que estão conectadas por uma única aresta não são comparadas no conjunto de blocos reestruturado.

Outra funcionalidade do *Reciprocal* CNP está relacionada aos vértices do grafo, em que um limiar de cardinalidade k é derivado ($k = \lfloor \sum_{b \in B} |b| / |P| - 1 \rfloor$). Este limiar determina o número máximo de arestas que serão mantidas e é utilizado como critério de poda da vizinhança de cada vértice. Uma estrutura de dados armazena os *top-k* vizinhos do vértice, considerando o peso da aresta que interliga o vértice ao vizinho analisado. Cinco esquemas foram propostos para a ponderação das arestas do grafo de blocagem: ARCS, CBS, ECBS, JS e EJS (mais detalhes em [Papadakis et al. 2014]). Todos normalizam os pesos entre [0, 1], de modo que os valores mais altos se referem às arestas que são mais propensas a conectar instâncias que se correspondem. A média de todos os esquemas de ponderação é utilizada para ponderar as arestas do grafo de blocagem.

4.2.2. Adaptação do Reciprocal CNP

A adaptação proposta neste trabalho para o *Reciprocal* CNP acontece exatamente na fase de ponderação. Segundo Shannon [2001], à medida que a ocorrência de um grupo de símbolos (termos) se torna mais frequente, a quantidade de informação decresce. Desta forma, foi utilizada para ponderar as arestas essa intuição, implementada por meio de uma função logarítmica, ou seja, se o par de instâncias (vértices interligados a aresta em análise) vier de um bloco cuja chave de blocagem for um termo que ocorre pouco na coleção, esse termo pode ser mais informativo (maior peso) do que outros, levando a manter no conjunto de blocos finais pares com maior probabilidade de correspondência.

A função $\frac{1}{log(x)}$ expressa bem essa questão, pois quanto menor o valor de x para valores positivos, maior o valor do resultado atribuído pela função. Assim, a ponderação das arestas é feita da seguinte forma: Dado uma frequência x de um termo (chave do bloco onde se extraiu a comparação entre p_i e p_j), é obtido o peso referente a aresta a_{ij}, sendo $peso(x) = \frac{1}{log(x)}$. Para cada aresta, é somado o peso encontrado anteriormente com o esquema de ponderação de arestas ECBS (*Enhanced Common Blocks*). A ponderação com o ECBS se dá da seguinte forma: $ECBS(p_i, p_j, B) = |B_{ij}| \times log\frac{|B|}{|B_i|} \times log\frac{|B|}{|B_j|}$, sendo $|B_{ij}|$ a quantidade de blocos comuns entre p_i e p_j, $|B_i|$ a quantidade de blocos que contém p_i, $|B_j|$ a quantidade de blocos que contém p_j e $|B|$ a quantidade total de blocos. Portanto, a equação final da ponderação de arestas é dada por: $pesoAresta(x, p_i, p_j, B) = \frac{1}{log(x)} + (|B_{ij}| \times log\frac{|B|}{|B_i|} \times log\frac{|B|}{|B_j|})$. Dessa forma, a técnica de processamento de blocos do PBBRT descarta comparações supérfluas, pela poda das arestas com peso menores, utilizando a frequência dos termos na coleção de dados para ponderar as arestas.

Tabela 1. Características técnicas das coleções de dados

	C1	C2	C3
|P|	63.869	50.797	3.354.773
|AV|	208.065	971.445	19.064.747
|D(P)|	2.308	22.863	892.579
||F||	2.580.284.412	1.290.142.206	5.627.249.263.378

5. Avaliação Experimental

A implementação do PBBRT foi feita em Java 8 como uma extensão do *framework* de código aberto apresentado em [Papadakis et al. 2014]. Todas as técnicas comparadas ao PBBRT neste trabalho foram implementadas no mesmo *framework*. Os experimentos foram realizados em um computador com processador Intel Xeon(R) E5620 2.40 GHz, 47GB de RAM e sistema operacional CentOS Linux 7. As medições de tempo de processamento de todas as técnicas foram repetidas 10 vezes e a média dos tempos é apresentada como resultado, com uma confiança de 95%, de modo a minimizar efeitos

5.1. Coleções de Dados

Para a avaliação experimental, foram utilizadas três coleções de dados semi-estruturados reais, que variam de tamanho e características. As coleções de dados referem-se a *Dirty ER*, ou seja, coleções de instâncias com duplicatas, disponíveis publicamente[1].

A Tabela 1 mostra características das coleções de dados para *Dirty ER*. A coleção C1 contém dados bibliográficos originados da DBLP (http://dblp.org) e Google Scholar (https://scholar.google.gr). A coleção C2 contém dados de filmes originados da IMDB (http://www.imdb.com) e da DBPedia (http://dbpedia.org). A coleção C3 contém instâncias de dois *snapshots* diferentes da *Wikipedia* em inglês (http://en.wikipedia.org). A seguinte notação é utilizada na apresentação das características técnicas das coleções de dados: $|P|$ representa o número de instâncias na coleção, $|AV|$ o número total de pares atributo-valor, $|D(P)|$ o número de duplicatas existentes, $||F||$ o número de comparações executadas pela abordagem força bruta, que compara cada instância com todas as outras.

5.2. Baseline

Os resultados deste trabalho são comparados com os apresentados por Papadakis et al. [2016]. Para a blocagem, Papadakis et al. [2016] utilizam a técnica *Token Blocking*. Em seguida, aplicam *Block Purging* [Papadakis et al. 2013] e *Block Filtering* [Papadakis et al. 2016]. *Block Purging* descarta os blocos que contêm mais da metade das instâncias da coleção, que são as chaves de blocagem altamente frequentes. *Block Filtering* visa reestruturar o conjunto de blocos eliminando as instâncias que são desnecessárias nos blocos. Essas tarefas são consideradas pré-processamento para a meta-blocagem, descartando mais da metade das arestas desnecessárias do grafo, em média.

Para o processamento de blocos, é utilizada para comparação a técnica de meta-blocagem *Reciprocal* CNP descrita na Subseção 4.2.1. Segundo Papadakis et al. [2016], as técnicas de meta-blocagem superam as demais técnicas de processamento de blocos.

5.3. Métricas de Avaliação

Para avaliar a qualidade de um conjunto de blocos B criado a partir do conjunto de instâncias da entrada P, são utilizadas as métricas *Pair Completeness (PC)* e *Pair Qua-*

[1]https://sourceforge.net/projects/erframework/files/DirtyERDatasets/RealDatasets/

lity (PQ) [Christen 2012]. Dado que $||B||$ é o número total de comparações nos blocos, $D(B)$ o conjunto de instâncias que coocorrem, $|D(B)|$ o seu tamanho (número de duplicatas possíveis de serem encontradas) e $|D(P)|$ o número de duplicatas existentes na coleção de dados (utiliza-se um gabarito onde as duplicatas estão identificadas), tem-se:

- *Pairs Completeness (PC)* é similar a revocação. Mede quão eficaz é a técnica em agrupar as duplicatas existentes. PC está definido no intervalo [0, 1], com valores mais altos indicando maior completude. Fórmula: $PC = \frac{|D(B)|}{|D(P)|}$.
- *Pairs Quality (PQ)* é similar a precisão. Mede quão eficiente é a técnica na obtenção dos blocos. PQ toma valores no intervalo [0, 1], com valores mais altos indicando maior qualidade para B. Fórmula: $PQ = \frac{|D(B)|}{||B||}$.

O desempenho da Resolução de Entidades pode ser distinguido em duas categorias: eficiência intensiva e eficácia intensiva. A eficiência intensiva tem como objetivo minimizar o tempo de processamento, sem deixar de detectar a maioria das duplicatas existentes. Mais formalmente, o seu objetivo é maximizar a qualidade dos pares (PQ) para uma completude (PC) que exceda 0,80. A eficácia intensiva permite um tempo de processamento mais elevado, desde que a completude (PC) seja maximizada, onde o seu valor não deve estar abaixo de 0,95 [Papadakis et al. 2016]. Juntamente com PC e PQ, outras duas métricas são utilizadas para a avaliação do processo de blocagem: $||B||$ indica o número total de comparações verdadeiras nos blocos; e o *Tempo de Processamento*, que mede o tempo necessário para extrair o conjunto de blocos finais. Para ambos, quanto menor o valor, melhor é o resultado. O intervalo de confiança (IC) para a média dos tempos de processamento serão apresentados usando uma distribuição *t de Student*.

5.4. Resultados

Dentre as características dos termos presentes em uma coleção avaliadas para identificar termos promissores para obter blocos, foram experimentadas a entropia (PC = 0,983 e PQ = 3,42E-04, coleção C1), frequência do termo na coleção (PC = 0,999 e PQ = 5,95E-05, coleção C1) e quantidade de caracteres do termo (PC = 0,999 e PQ = 3,23E-05, coleção C1). Entropia teve resultados melhores para C1 e também para C2 e C3, em termos de PQ, levando a sua escolha. Por brevidade, são apresentados os resultados apenas das características que obtiveram os melhores resultados.

A Tabela 2 mostra o desempenho da blocagem do PBBRT (representado por PBBRT P1, primeira parte do processo do PBBRT), em comparação com a técnica de blocagem usada em [Papadakis et al. 2016] (representado por T), aplicados às coleções de dados C1, C2 e C3. Observa-se que o PBBRT P1 obteve uma redução no número de comparações nos blocos de 68% em média, considerando as três coleções de dados, ao custo de uma redução de PC em torno de 1,7%, mantendo, ainda assim, o PC acima de 0,95 para todas as coleções. Dessa forma, PQ aumenta em média 4 vezes, diminuindo pela metade o tempo de execução nas coleções C1 e C2 e em 64% para coleção C3. Com a técnica de blocagem do PBBRT escolheu-se os termos relevantes, criando, assim, um número menor de blocos e tornando o processo de blocagem mais rápido e preciso.

No entanto, resultados mais satisfatórios para a precisão foram encontrados com a aplicação da técnica de processamento de blocos. A Tabela 3 mostra os resultados encontrados com a primeira e a segunda parte do PBBRT (blocagem + processamento de blocos), comparados ao *Reciprocal* CNP aplicado aos blocos criados, apresentados em Papadakis et al. [2016]. Os resultados mostrados para o *Reciprocal* CNP utilizam para

Tabela 2. Comparações das técnicas de blocagem

	C1		C2		C3	
	T	**PBBRT P1**	**T**	**PBBRT P1**	**T**	**PBBRT P1**
PC	0,994	0,983	0,976	0,951	0,997	0,982
PQ	9,62E-05	3,42E-04	1,62E-04	1,08E-03	3,86E-05	7,29E-05
||B||	2,38E+07	6,64E+06	1,37E+08	2,02E+07	2,31E+10	1,20E+10
Tempo	4,3 s	2,2 s	8,4 s	4,2 s	13 min	4,7 min
IC Tempo	± 0,05 s	± 0,1 s	± 0,2 s	± 0,1 s	± 11,7 s	± 5,6 s

ponderação de arestas a média dos cinco esquemas propostos em Papadakis et al. [2014] (ARCS, CBS, ECBS, JS e EJS). Porém, para a técnica de processamento de blocos do PBBRT somente o ECBS foi utilizado. Assim, são comparados ao PBBRT, o *Reciprocal* CNP usando a média dos cinco esquemas de ponderação de arestas (representado por M1) e o *Reciprocal* CNP com o ECBS (representado por M2).

Tabela 3. Comparações das técnicas de processamento de blocos

	C1			C2			C3		
	M1	**M2**	**PBBRT**	**M1**	**M2**	**PBBRT**	**M1**	**M2**	**PBBRT**
PC	0,846	0,867	0,855	0,650	0,736	0,760	0,868	0,882	0,871
PQ	0,017	0,016	0,024	0,057	0,063	0,078	0,111	0,102	0,132
||B||	1,19E+05	1,25E+05	8,35E+04	2,86E+05	2,65E+05	2,23E+05	7,12E+06	7,73E+06	5,87E+06
Tempo	22,8 s	21,9 s	10,8 s	8,1 min	5,0 min	57,8 s	13,9 h	10,6 h	7,9 h
IC Tempo	± 0,5 s	± 0,5 s	± 0,4 s	± 12,4 s	± 11,2 s	± 0,9 s	± 1,8 min	± 1,7 min	± 12,9 s

Comparado à técnica M1, o PBBRT tem melhores resultados sob todas as métricas de avaliação em todas as coleções de dados. O número de comparações diminui em média 23,4%, com um aumento no PC de 16,9% na coleção C2 e em torno de 1% nas coleções C1 e C3. Em média, PQ aumenta em torno de 32,3% e o tempo de processamento é reduzido em torno de 61,2%. A técnica M2 comparada ao PBBRT só ganha em relação ao PC nas coleções C1 e C3, em torno de 1,3%. Vale ressaltar, que o PC desejado deve estar acima de 0,80 para eficiência intensiva. Na coleção C2, esse valor para a métrica PC não é atingido por nenhuma das técnicas em comparação. Porém, o PBBRT melhora o resultado para PC em relação a M1 e M2.

Apesar da revocação (PC) variar muito pouco comparando as técnicas, há melhorias na precisão (PQ) e ganhos expressivos em relação ao tempo total de processamento. Os tempos de processamento das técnicas mostrados na Tabela 2 e na Tabela 3 foram analisados utilizando o teste de hipótese *t de Student*, avaliando se havia diferença significativa entre as médias dos tempos de processamento das técnicas. A hipótese nula, que afirma que as duas médias de tempos são iguais, foi rejeitada para todas as técnicas em todas as coleções de dados com uma confiança de 95%, comprovando o ganho em relação ao tempo de processamento do PBBRT.

6. Conclusões

Neste trabalho, foram apresentadas melhorias no processo de blocagem por meio do PBBRT. O PBBRT verifica a relevância dos termos presentes em coleções de dados e utiliza tais informações com o objetivo de aumentar a qualidade dos blocos, diminuindo o número de comparações em uma tarefa de Resolução de Entidades, por exemplo. O PBBRT foi avaliado experimentalmente em coleções de dados reais e os resultados mostram que o PBBRT supera uma técnica representativa de meta-blocagem em até 16,9% de

completude e em 32,3%, em média, na qualidade dos blocos gerados, reduzindo o tempo de processamento aproximadamente pela metade. Assim, foi demonstrado que o PBBRT pode processar eficientemente grandes coleções de dados altamente heterogêneas.

Como trabalhos futuros, pretende-se adaptar o PBBRT para a tarefa *Clean-Clean ER*, avaliar outras características baseada em termos e outros meios de ponderação de arestas, visando melhorar ainda mais os resultados em termos de eficiência e eficácia.

Agradecimentos. Este trabalho foi apoiado e financidao pela Universidade Federal de Ouro Preto (UFOP), Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq (grant 312395/2017-5).

Referências

Christen, P. (2012). A survey of indexing techniques for scalable record linkage and deduplication. *IEEE TKDE*, 24(9):1537–1555.

Fellegi, I. P. and Sunter, A. B. (1969). A theory for record linkage. volume 64, pages 1183–1210.

Halevy, A., Rajaraman, A., and Ordille, J. (2006). Data integration: the teenage years. In *VLDB*, pages 9–16.

Hernández, M. A. and Stolfo, S. J. (1995). The merge/purge problem for large databases. *ACM SIGMOD Rec.*, 24(2):127–138.

Madhavan, J., Jeffery, S. R., Cohen, S., Dong, X., Ko, D., Yu, C., and Halevy, A. (2007). Web-scale data integration: You can only afford to pay as you go. In *CIDR*, pages 342–350.

McCallum, A., Nigam, K., and Ungar, L. H. (2000). Efficient clustering of high-dimensional data sets with application to reference matching. In *ACM SIGKDD*, pages 169–178.

Papadakis, G., Ioannou, E., Palpanas, T., Niederee, C., and Nejdl, W. (2013). A blocking framework for entity resolution in highly heterogeneous information spaces. *IEEE TKDE*, 25(12):2665–2682.

Papadakis, G., Koutrika, G., Palpanas, T., and Nejdl, W. (2014). Meta-blocking: Taking entity resolutionto the next level. *IEEE TKDEFherna*, 26(8):1946–1960.

Papadakis, G., Papastefanatos, G., Palpanas, T., and Koubarakis, M. (2016). Scaling entity resolution to large, heterogeneous data with enhanced meta-blocking. In *EDBT*, pages 221–232.

Shannon, C. E. (2001). A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55.

Simonini, G., Bergamaschi, S., and Jagadish, H. (2016). Blast: a loosely schema-aware meta-blocking approach for entity resolution. *VLDB*, 9(12):1173–1184.

Whang, S. E., Menestrina, D., Koutrika, G., Theobald, M., and Garcia-Molina, H. (2009). Entity resolution with iterative blocking. In *ACM SIGMOD*, pages 219–232.

Wilbur, W. J. and Sirotkin, K. (1992). The automatic identification of stop words. *Journal of information science*, 18(1):45–55.

Finding Top-k Sequences over Data Streams according to Temporal Conditional Preferences

Marcos Roberto Ribeiro[1,2], Maria Camila N. Barioni[2], Sandra de Amo[2], Claudia Roncancio[3], Cyril Labbé[3]

[1] Instituto Federal de Minas Gerais (IFMG), Bambuí, Brazil

[2]Universidade Federal de Uberlândia (UFU), Uberlândia, Brazil

[3]Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG, F-38000, Grenoble, France

marcos.ribeiro@ifmg.edu.br, {camila.barioni, deamo}@ufu.br,
{claudia.roncancio, cyril.labbe}@imag.fr

Abstract. *Recently, several research works have been conducted on processing of preference queries over data streams. Preference queries are useful for many application domains where users aim to find out the closest data items to their wishes. This paper presents a new operator for the StreamPref language that can be employed to obtain the top-k data stream sequences according to temporal conditional preferences. Temporal conditional preferences can allow a user to express how past instants of a data stream may influence his preferences at a present instant. In order to evaluate this new operator, two new algorithm strategies are also presented. The extensive set of experiments performed show that the incremental strategy presents a superior performance in all experimental settings. Moreover, the results achieved show that the proposed operator has a superior performance when compared to the equivalent operation in CQL.*

1. Introduction

Preference queries aim to select the closest data items to the user wishes [Ribeiro et al. 2016]. Considering posing this type of query in data stream scenarios, preference queries must be processed efficiently to meet the high-speed data transfer requirement. There are several related research works concerned with skyline queries where the user preferences are represented as wishes for maximum or minimum attribute values [Börzsönyi et al. 2001, Lin et al. 2005, Tao and Papadias 2006]. However, there are application domains that require the users to express conditional preferences. This kind of preference allows the user to say how some attribute value influences their preferences over another attribute. In order to illustrate, let us consider a soccer coach who wants to hire a player based on his nationality. He can express his desire considering a conditional preference as follows: "if the player is Brazilian then I prefer the attack position than the midfield position".

The data tuples in data stream scenarios have an implicit temporal information. If we consider this rich information, it is possible to use conditional temporal preferences to express how a user wishes at a given moment are impacted by past attribute values. As an example, consider a coach who wants to monitor a data stream of a soccer match. The coach may use preferences such as "if the player was in offensive intermediary then I prefer that he stays in the same place instead of going to midfield". We have been exploring

this research topic in some preliminary works. We proposed a new formalism for reasoning with temporal conditional preferences [Ribeiro et al. 2017a] and used this formalism to define the first version of the StreamPref query language which allows querying data streams with temporal conditional preferences [Ribeiro et al. 2017b].

The current version of the StreamPref query language is able to select dominant sequences. A sequence s is dominant if there is no other sequence better than s. However, if a query returns few dominant sequences, this result could not be enough for the user. The user may want to rank the sequences according to their preferences to get the best k sequences in this rank (i.e., the top-k sequences). For instance, the same soccer coach mentioned previously can be also interested in answering the following query: "Give me the best four sequences of positioning according to my preferences". Others interesting practical applications for queries with temporal preferences are stock market, telecommunications, web applications, sensor networks, among others. This paper presents a new operator that incorporates the ability to select the top-k sequences according to temporal conditional preferences in the StreamPref query language.

The main contributions of this paper can be summarized as follows: **(1)** We propose an extension of the StreamPref query language with a new operator that allows to find top-k sequences according to temporal conditional preferences; **(2)** We present the demonstration of the equivalence for the proposed operator and the existing operators; **(3)** We propose an efficient algorithm to evaluate the new operator; **(4)** We describe the results of an extensive set of experiments comparing two strategies used by our algorithm.

The remainder of this paper is organized as follows. Section 2 introduces the logical formalism and the existing operators of the StreamPref language. Section 3 presents our new proposed operator to extend the StreamPref language. Section 4 describes the algorithm used to evaluate the proposed operator. Section 5 presents the experiments and discusses the results. Section 6 discusses the main related research works. Finally, the conclusion and the future work directions are presented in Section 7.

2. The StreamPref Language

This section presents the fundamental concepts regarding the preference model and the operators of the StreamPref query language [Ribeiro et al. 2017a, Ribeiro et al. 2017b]. Section 2.1 describes the preference model and Section 2.2 presents the existing operators.

2.1. Temporal Conditional Preferences

Let $R(A_1, ..., A_l)$ be a relational schema. A sequence $s = \langle t_1, ..., t_n \rangle$ over R is an ordered set of tuples, such that $t_i \in \mathbf{Tup}(R)$ for all $i \in \{1, ..., n\}$ where $\mathbf{Tup}(R) = \mathbf{Dom}(A_1) \times ... \times \mathbf{Dom}(A_l)$ is the set of all tuples over R. The length of a sequence s is denoted by $|s|$. A tuple in the position i of a sequence s is denoted by $s[i]$ while the notation $s[i].A$ represents the attribute A in the position i of s. The set of all possible sequences over R is denoted by $\mathbf{Seq}(R)$. The StreamPref formulas are based on propositions $(A\theta a)$, where $a \in \mathbf{Dom}(A)$ and $\theta \in \{<, \leq, =, \neq, \geq, >\}$ (see Definition 1). Let $Q(A)$ be a proposition, $S_{Q(A)} = \{a \in \mathbf{Dom}(A) \mid a \models Q(A)\}$ denotes the set of values satisfying $Q(A)$.

Definition 1 (StreamPref Formulas) *The StreamPref formulas are defined as follows:* ***(1)*** $\mathbf{true}$ *and* $\mathbf{false}$ *are StreamPref formulas;* ***(2)*** *If F is a proposition then F is a Stream-*

Pref formula; ***(3)*** *If F and G are StreamPref formulas then $(F \wedge G)$, $(F \vee G)$, $(F \ \mathbf{since}\ G)$, $\neg F$ and $\neg G$ are StreamPref formulas.*

A StreamPref formula F is satisfied by a sequence $s = \langle t_1, ..., t_n \rangle$ at a position $i \in \{1, ..., n\}$, denoted by $(s, i) \models F$, according to the following conditions: ***(1)*** $(s, i) \models Q(A)$ if and only if $s[i].A \models Q(A)$; ***(2)*** $(s, i) \models F \wedge G$ if and only if $(s, i) \models F$ and $(s, i) \models G$; ***(3)*** $(s, i) \models F \vee G$ if and only if $(s, i) \models F$ or $(s, i) \models G$; ***(4)*** $(s, i) \models \neg F$ if and only if $(s, i) \not\models F$; ***(5)*** $(s, i) \models (F \ \mathbf{since}\ G)$ if and only if there exists j where $1 \leq j < i$ and $(s, j) \models G$ and $(s, k) \models F$ for all $k \in \{j + 1, ..., i\}$. The $\mathbf{true}$ formula is always satisfied and the $\mathbf{false}$ formula is never satisfied. The StreamPref also has the following derived formulas:

Prev $Q(A)$**:** Equivalent to $(\mathbf{false\ since}\ Q(A))$, $(s, i) \models \mathbf{Prev}\ Q(A)$ if and only if $i > 1$ and $(s, i - 1) \models Q(A)$;

SomePrev $Q(A)$**:** Equivalent to $(\mathbf{true\ since}\ Q(A))$, $(s, i) \models \mathbf{SomePrev}\ Q(A)$ if and only if there exists j such that $1 \leq j < i$ and $(s, j) \models Q(A)$;

AllPrev $Q(A)$**:** Equivalent to $\neg(\mathbf{SomePrev} \neg Q(A))$, $(s, i) \models \mathbf{AllPrev}\ Q(A)$ if and only if $(s, j) \models Q(A)$ for all $j \in \{1, ..., i - 1\}$;

First: Equivalent to $\neg(\mathbf{Prev}(\mathbf{true}))$, $(s, i) \models \mathbf{First}$ if and only if $i = 1$.

Definition 2 formalizes the *temporal conditions* used by Definition 3 (*tcp-rules* and *tcp-theories*). Example 1 shows a practical application using these definitions.

Definition 2 (Temporal Conditions) *A temporal condition is a formula $F = F_1 \wedge ... \wedge F_n$, where $F_1, ..., F_n$ are propositions or derived formulas. The temporal components of F, denoted by $F^{\leftarrow}$, are the conjunction of all derived formulas in F. The non-temporal components of F, denoted by $F^{\bullet}$, is the conjunction of all propositions in F and not present in $F^{\leftarrow}$. The notation $\mathbf{Att}(F)$ represents the attributes appearing in F.*

Definition 3 (TCP-Rules and TCP-Theories) *Let R be a relational schema. A temporal conditional preference rule, or tcp-rule, is an expression in the format $\varphi : C_\varphi \rightarrow Q^+_\varphi \succ Q^-_\varphi[W_\varphi]$, where:* ***(1)*** *The propositions Q^+_φ and Q^-_φ, over the preference attribute A_φ, represent the preferred values and non-preferred values, respectively, such that $S_{Q^+_\varphi} \cap S_{Q^-_\varphi} = \{\}$;* ***(2)*** *$W_\varphi \subset R$ is the set of indifferent attributes such that $A_\varphi \notin W_\varphi$;* ***(3)*** *$C_\varphi$ is a temporal condition such that $\mathbf{Att}(C^{\bullet}_\varphi) \cap \{A_\varphi\} \cap W_\varphi = \{\}$. A temporal conditional preference theory, or tcp-theory, is a finite set of tcp-rules.*

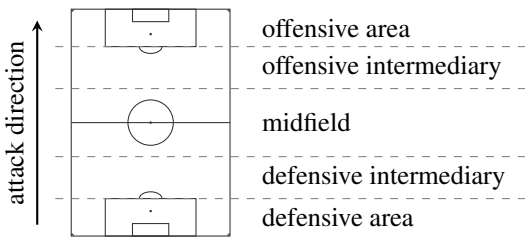


Figure 1. Soccer field places

Example 1 *Suppose a soccer coach who has access to an information system that provides real-time data concerning field positioning of the players. The data is in*

the stream $\texttt{positioning}(\underline{\texttt{pid}}, \texttt{place}, \texttt{ball}, \texttt{direction})$ *composed of the attributes* `pid` *(player identifier),* `place` *(field place),* `ball` *(ball possession) and* `direction` *(moving direction). The field places are depicted in Figure 1. The attribute* `ball` *has the value* 1 *when the team has the ball possession and* 0 *for otherwise. For the attribute* `direction`*, the possible values are forward (fw), lateral (la) and rewind (rw). The coach has the following preferences:* ***[P1]*** *Lateral moves are better than forward moves, independent of ball possession;* ***[P2]*** *Forward moves are better than rewind moves, independent of ball possession;* ***[P3]*** *If, in a given moment, the team does not have the ball possession and, immediately before this moment, the player was at offensive intermediary and, always before this moment, the team had the ball, then I prefer midfield place than offensive intermediary place;* ***[P4]*** *If, in a given moment, the team has the ball possession and, immediately before this moment, the player was at midfield then I prefer offensive intermediary place than midfield place; These preferences can be expressed using the tcp-theory* Φ *composed of the following tcp-rules:* $\varphi_1 : \rightarrow (\texttt{direction} = la) \succ (\texttt{direction} = fw)[\texttt{ball}]$; $\varphi_2 : \rightarrow (\texttt{direction} = fw) \succ (\texttt{direction} = rw)[\texttt{ball}]$; $\varphi_3 : (\texttt{ball} = 0) \wedge \mathbf{Prev}(\texttt{place} = oi) \wedge \mathbf{AllPrev}(\texttt{ball} = 1) \rightarrow (\texttt{place} = mf) \succ (\texttt{place} = oi)$; $\varphi_4 : (\texttt{ball} = 1) \wedge \mathbf{Prev}(\texttt{place} = mf) \rightarrow (\texttt{place} = oi) \succ (\texttt{place} = mf)$.

A sequence s is preferred to a sequence s' (or s dominates s') according to a φ, denoted by $s \succ_\varphi s'$, if and only if there exists a position i such that: ***(1)*** All positions before i must be identical in both sequences, $s[j] = s'[j]$ for all $j \in \{1, ..., i-1\}$; ***(2)*** The position i of s and s' must satisfy the rule condition C_φ, $(s, i) \models C_\varphi$ and $(s', i) \models C_\varphi$; ***(3)*** The position i of s has a preferred value and the position i of s' has a non-preferred value, $s[i].A_\varphi \models Q^+_\varphi$ and $s'[i].A_\varphi \models Q^-_\varphi$; ***(4)*** Excluding the preference attribute A_φ and the indifferent attributes of W_φ, all attributes of position i must have identical values in both sequences (*ceteris paribus* semantic), $s[i].A' = s'[i].A'$ for all $A' \notin (\{A_\varphi\} \cup W_\varphi)$. Example 2 presents a possible comparison of sequences.

Example 2 *Considering the sequences* $s = \langle(oi, 1, la), (oi, 1, fw), (oi, 0, fw)\rangle$ *and* $s' = \langle(oi, 1, la), (oi, 1, fw), (mf, 0, fw)\rangle$ *and the tcp-theory* Φ *of Example 1. It is possible to say that* $s \succ_{\varphi_3} s'$ *since:* ***(1)*** $s[1] = s'[1]$ *and* $s[2] = s'[2]$; ***(2)*** $(s, 2) \models (\texttt{ball} = 0) \wedge \mathbf{Prev}(\texttt{place} = oi) \wedge \mathbf{AllPrev}(\texttt{ball} = 1)$ *and* $(s', 2) \models (\texttt{ball} = 0) \wedge \mathbf{Prev}(\texttt{place} = oi) \wedge \mathbf{AllPrev}(\texttt{ball} = 1)$; ***(3)*** $s[3].\texttt{place} = mf$ *(preferred value),* $s'[3].\texttt{place} = oi$ *(non preferred value);* ***(4)*** $s[3].\texttt{ball} = s'[3].\texttt{ball}$ *and* $s[3].\texttt{direction} = s'[3].\texttt{direction}$ *(ceteris paribus semantic).*

The notation $\succ_\Phi$ represents the transitive closure of $\bigcup_{\varphi \in \Phi} \succ_\varphi$. Let Φ be a tcp-theory over a relational schema $\mathbf{Seq}(R)$. A sequence $s \in \mathbf{Seq}(R)$ is preferred to $s' \in \mathbf{Seq}(R)$ according to Φ, denoted by $s \succ_\Phi s'$, if there exists the sequences $s_1, ..., s_{m+1} \in \mathbf{Seq}(R)$ and the tcp-rules $\varphi_1, ..., \varphi_m \in \Phi$ such that $s_1 \succ_{\varphi_1} ... \succ_{\varphi_m} s_{m+1}$, where $s = s_1$ and $s' = s_{m+1}$. When two sequences cannot be compared, they are called incomparable. For instance, consider the sequences s and s' of Example 2 and the sequence $s'' = \langle(oi, 1, la), (oi, 1, fw), (mf, 1, la)\rangle$. We have the comparisons $s \succ_{\varphi_3} s'$ and $s' \succ_{\varphi_1} s''$. So, by transitivity, $s \succ_\Phi s''$. We must also consider consistency issues when dealing with order induced by rules to avoid inferences such as "a sequence is preferred to itself". Please, see [Ribeiro et al. 2017a] for more details about consistency issues.

2.2. StreamPref Operators

The first step in the evaluation of a continuous tcp-query is the extraction of sequences using **SEQ** operator. This task is performed by the operation $\mathbf{SEQ}_{X,n,d}(S)$, where X is the set of identifier attributes, n is the temporal range, d is the slide interval and S is the input data stream. The parameters n and d are used to delimit a portion of the data stream analogously to the selection performed by the *sliding window* operators [Arasu et al. 2016]. The parameter X is a key used to group the tuples with the same identifier in a sequence. Example 3 demonstrates the use of the **SEQ** operator.

Example 3 *Consider the stream* `positioning (pid, place, ball, direction)` *of Figure 2(a) and the preferences of Example 1. Now, suppose that a coach submits the following query to the information system: "**[Q1]** At every instant, give me the sequences of positioning that best fit my preferences over the last 3 seconds". The extraction of sequences is performed by the operation* $\mathbf{SEQ}_{\{pid\},3,1}(positioning)$. *Figure 2(b) shows, instant by instant, the result of this operation. As the user wants to consider just the last three seconds, from instant 3, the old tuples are removed from the sequences.*

Instant	pid	place	ball	direction
0	1	mf	1	la
0	2	oi	1	la
0	3	mf	1	fw
0	4	oi	1	la
0	5	oi	1	la
1	1	oi	0	la
1	2	oi	0	la
1	3	mf	0	la
1	4	mf	0	la
1	5	oi	0	fw
2	1	oi	1	la
2	2	oi	1	rw
2	3	di	1	la
2	4	di	1	rw
2	5	oi	1	rw
3	1	oi	0	rw
3	2	oi	0	rw
3	3	mf	0	la
3	4	oi	0	rw
3	5	mf	0	rw

(a)

Instant 0
- $s_1 = \langle(mf, 1, la)\rangle$
- $s_2 = \langle(oi, 1, la)\rangle$
- $s_3 = \langle(mf, 1, fw)\rangle$
- $s_4 = \langle(oi, 1, la)\rangle$
- $s_5 = \langle(oi, 1, la)\rangle$

Instant 1
- $s_1 = \langle(mf, 1, la), (oi, 0, la)\rangle$
- $s_2 = \langle(oi, 1, la), (oi, 0, la)\rangle$
- $s_3 = \langle(mf, 1, fw), (mf, 0, la)\rangle$
- $s_4 = \langle(oi, 1, la), (mf, 0, la)\rangle$
- $s_5 = \langle(oi, 1, la), (oi, 0, fw)\rangle$

Instant 2
- $s_1 = \langle(mf, 1, la), (oi, 0, la), (oi, 1, la)\rangle$
- $s_2 = \langle(oi, 1, la), (oi, 0, la), (oi, 1, rw)\rangle$
- $s_3 = \langle(mf, 1, fw), (mf, 0, la), (di, 1, la)\rangle$
- $s_4 = \langle(oi, 1, la), (mf, 0, la), (di, 1, rw)\rangle$
- $s_5 = \langle(oi, 1, la), (oi, 0, fw), (oi, 1, rw)\rangle$

Instant 3
- $s_1 = \langle\cancel{(mf, 1, la)}, (oi, 0, la), (oi, 1, la), (oi, 0, rw)\rangle$
- $s_2 = \langle\cancel{(oi, 1, la)}, (oi, 0, la), (oi, 1, rw), (oi, 0, rw)\rangle$
- $s_3 = \langle\cancel{(mf, 1, fw)}, (mf, 0, la), (di, 1, la), (mf, 0, la)\rangle$
- $s_4 = \langle\cancel{(oi, 1, la)}, (mf, 0, la), (di, 1, rw), (oi, 0, rw)\rangle$
- $s_5 = \langle\cancel{(oi, 1, la)}, (oi, 0, fw), (oi, 1, rw), (mf, 0, rw)\rangle$

(b)

Figure 2. (a) Stream `positioning` (b) Sequences extracted by SEQ operator.

Let Z be a set of sequences and Φ be a tcp-theory. The operation $\mathbf{BESTSEQ}_{\Phi}(Z)$ returns the *dominant* sequences in Z according to Φ. A sequence $s \in Z$ is dominant according to Φ, if $\nexists s' \in Z$ such that $s' \succ_{\Phi} s$. Example 4 shows how the **BESTSEQ** operator can be used to evaluate a query.

Example 4 *Let Z be the extracted sequences from Example 3 at instant 3. We can select the best sequences according to the tcp-theory in Example 1 by using the operation* $\mathbf{BESTSEQ}_{\Phi}(\mathbf{SEQ}_{\{pid\},3,1}(positioning))$. *At instant 3, $s_1 \succ_{\Phi} s_2$, $s_1 \succ_{\Phi} s_5$, $s_2 \succ_{\Phi} s_5$ and $s_3 \succ_{\Phi} s_4$. Thus, the result of query **Q1** at instant 3 is $\{s_1, s_3\}$.*

3. The New Operator TOPKSEQ

It is possible to use the **BESTSEQ** operator to obtain the top-k sequences. However, as we will see at the end of this section, the **BESTSEQ** operator is not suitable for this task. Thus, in this paper, we propose the **TOPKSEQ** operator in order to select the top-k sequences. The **TOPKSEQ** operator returns the top-k sequences of an input set of sequences Z according to a tcp-theory Φ. The top-k sequences are the sequences of Z with the lowest preference level (Definition 4)

Definition 4 (Preference level) *Let Φ be a tcp-theory. Let Z be a set of sequences. The preference level of a sequences s, denoted by $level(s)$, is:* ***(1)*** *If $\nexists s' \in Z$ such that $s' \succ_\Phi s$, then $level(s) = 0$;* ***(2)*** *Otherwise, $level(s) = \max\{level(s') \mid s' \in Z$* ***and*** *$s' \succ_\Phi s\} + 1$.*

Notice that the sequences with level zero are exactly those returned by the **BESTSEQ** operator. The **TOPKSEQ** operator is especially useful when the result of the **BESTSEQ** operator has few sequences. In this case, the **TOPKSEQ** operator can complement the answer using sequences with greater levels (see Example 5).

Example 5 *Consider again the Example 4. The query result has just two sequences. Now, suppose that the coach has the following query: "**[Q2]** At every instant, give me the best four sequences of positioning according to my preferences over the last 3 seconds". This query is evaluated by the operation $\mathbf{TOPKSEQ}_{\Phi,4}(\mathbf{SEQ}_{\{pid\},3,1}(positioning))$. At instant 3, $s_1 \succ_\Phi s_2$, $s_1 \succ_\Phi s_5$, $s_2 \succ_\Phi s_5$ and $s_3 \succ_\Phi s_4$. So, the preference levels are $level(s_1) = 0$, $level(s_3) = 0$, $level(s_2) = \max\{level(s_1)\} + 1 = 1$, $level(s_4) = \max\{level(s_3)\} + 1 = 1$ and $level(s_5) = \max\{level(s_1), level(s_2)\} + 1 = 2$. Thus the result of query **Q2** at instant 3 is $\{s_1, s_3, s_2, s_4\}$.*

The StreamPref language is an extension of the Continuous Query Language (CQL). The StreamPref operators do not increase the expression power of the CQL [Ribeiro et al. 2017b]. However, the equivalences are not trivial since the StremPref operators is equivalent to complex operations using several CQL operators and intermediary relations. So, these equivalences are not simple to be written by the user. Moreover, the new operator has algorithms that are specially tailored to process queries more efficiently than their CQL counterparts. The CQL equivalences for the SEQ and **BESTSEQ** operators were already demonstrated in our previous work [Ribeiro et al. 2017b]. Equations (1a)-(1d) show how we can use the **BESTSEQ** operator to evaluate the **TOPKSEQ** operator. As the **BESTSEQ** operator has a CQL equivalence, we can conclude that the **TOPKSEQ** operator has also a CQL counterpart.

$$L_0 \leftarrow \mathbf{BESTSEQ}_\Phi(Z) \tag{1a}$$
$$L_1 \leftarrow \mathbf{BESTSEQ}_\Phi(Z - L_0) \tag{1b}$$
$$L_2 \leftarrow \mathbf{BESTSEQ}_\Phi(Z - L_1 - L_0) \tag{1c}$$
$$\vdots$$
$$L_m \leftarrow \mathbf{BESTSEQ}_\Phi(Z - L_{m-1} - ... - L_0) \tag{1d}$$

The term m of Equation (1d) represents the maximum preference level imposed by Φ. This number is equal to the number of tcp-rules of Φ in the worst case. Each set L_i contains the sequences with preference level i. The top-k sequences can be obtained by

taking the sequences of these sets (following the preference level order) until k sequences are reached. Despite the TOPKSEQ operator can be evaluated using the BESTSEQ operator, it is necessary to process all sequences at every instant to reach all preference levels and sort the sequences by level. On the other hand, Section 4 presents algorithms which stop the processing after the top-k sequences are obtained.

4. The Algorithm

As discussed in the previous section, the $\mathbf{TOPKSEQ}$ operator can be processed by using the $\mathbf{BESTSEQ}$ operator a certain number of times. The algorithm *GetTopkSeq* (see Algorithm 1) employs this idea to evaluates the $\mathbf{TOPKSEQ}$ operator. First, the algorithm creates a list to keep the sequences ordered by their preference level. The first iteration of the loop uses the *GetBestSeq* routine to select the sequences with level zero. This routine basically removes the dominant sequences from Z. Every iteration of the loop selects the sequences of the next level. This process stops when the list has at least k sequences or Z is empty.

Algorithm 1: *GetTopkSeq*(Φ, k, Z)

```
1 L ← NewList();
2 while (|L| < k) and Z ≠ {} do
3 |   Z' ← GetBestSeq(Φ, Z);
4 |   L.append(Z');
5 return L.getFirst(k);
```

Algorithm 2: *NaiveBestSeq*(Φ, Z)

```
1 Z' ← Z;
2 foreach s, s' ∈ Z' do
3 |   if s ≻_Φ s' then Z' ← Z' − {s'} ;
4 |   else if s' ≻_Φ s then Z' ← Z' − {s} ;
5 return Z';
```

The *GetBestSeq* routine is basically an algorithm to evaluate the $\mathbf{BESTSEQ}$ operator. This algorithm can use a naive approach [Ribeiro et al. 2017a] or an incremental approach [Ribeiro et al. 2017b]. The naive approach must compare all sequences at every instant as addressed by the algorithm *NaiveBestSeq* (see Algorithm 2). On the other hand, the incremental approach keeps an index structure updated using just the sequence changes. This index structure is a sequence tree created using the sequences tuples.

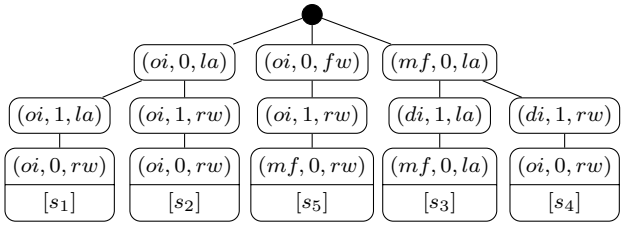


Figure 3. Sequence tree

Figure 3 shows the sequence tree built with the sequences shown in Example 3 at instant 3. Only the changed sequences are reallocated in the tree. The tree structure is useful to find the position where two sequences must be compared (the fork nodes). In addition, a node keeps a preference hierarchy representing the children comparison. So, for each tree node, we can obtain the dominant nodes (and, by consequence, the dominant sequences) by using this preference hierarchy. Please see [Ribeiro et al. 2017b] for more details about the index structure. The algorithm *IncBestSeq* (see Algorithm 3) obtains the dominant sequences by using the sequence tree. The algorithm starts at the tree root and uses recursive calls over the dominant children to reach all dominant sequences.

Algorithm 3: *IncBestSeq*(nd)

```
1 Z ← nd.Z;
2 foreach dominant child of nd do
3 |   Z ← Z ∪ IncBestSeq(child);
4 return Z;
```

The naive algorithm must find the position to be compared. In the incremental version, the sequence tree already points to the position to be compared. In addition, the tree nodes use preference hierarchies to store many comparisons of previous instants. Only changed sequences cause updates in the tree and preference hierarchy.

The complexity analysis of the algorithms takes into account the number of input sequences (z), the length of the largest sequence (n) and the number of tcp-rules in Φ (m). We assume a constant factor for the number of attributes. The algorithm *NaiveBestSeq* must compare every pair of sequences. The comparison start by looking for the first different position in the sequences. In the worst case, this position is the last one (n). Next, for the dominance test, the algorithm uses a deep first search strategy to find a chain of sequences and rules. The search tree of this strategy has height and node degree equal to m in the worst case. Thus, the complexity of the algorithm *NaiveBestSeq* is $O(z^2 \times (n + m^m))$ where the factor m^m is the cost of the dominance test.

The incremental strategy to obtain the dominant sequences must update the sequence tree. In the worst case scenario, the degree of nodes is $O(z)$ and the tree depth is $O(n)$. We also have to consider the cost to deal with the preference hierarchy. Our preference hierarchy uses the partition strategy described in [Ribeiro et al. 2016]. The update cost of this hierarchy is m^4. Thus, the complexity of *IncBestSeq* is $O(zn \times m^4)$ since every sequence can cause the update of n nodes.

The cost of the algorithm *GetTopkSeq* is related to the complexity and number of calls to the routine *GetBestSeq*. This routine is called $O(m)$ times in the worst case. Thus, the complexity of the algorithm *GetTopkSeq* is the cost of this routine multiplied by m. In data stream scenarios, the incremental algorithms usually are faster than naive algorithms. The experimental results of the next section show this tendency in the algorithms when processing the $\mathbf{TOPKSEQ}$ operator.

5. Experimental Results

We conducted an extensive set of experiments to analyze the performance (runtime) and the memory usage of the algorithms used to evaluate the $\mathbf{TOPKSEQ}$ operator. All experiments were carried out on a machine with a 3.2 GHz twelve-core processor and 32 GB of main memory, running Linux. The algorithms were implemented in a Data Stream Management System (DSMS) prototype using Python language[1].

The same tool used in [Ribeiro et al. 2017b] was employed to generate the synthetic datasets for our experiments[2]. This tool generates streams composed of integer attributes. In addition, it allows evaluating several parameter settings. For each experi-

[1] `http://streampref.github.io/`

[2] `http://streampref.github.io/streamprefgen/`

ment, we varied one parameter and fixed a default value for the others. We do not include the CQL equivalence in our experiments because this equivalence was already discussed in [Ribeiro et al. 2017b]. The experimental results described in [Ribeiro et al. 2017b] showed higher runtime and greater memory usage for the CQL equivalence due to its various intermediary operation and temporary relations.

Table 1. The parameters of the experiments: (a) Data generation; (b) Sequence extraction; (c) Preferences.

(a)		(b)		(c)	
Param.	Variation	Param.	Variation	Param.	Variation
ATT	8, 10, **12**, 14, 16	RAN	20, 40, **60**, 80, 100	RUL	8, 16, **24**, 32, 40
NSQ	8, 16, **24**, 32, 40	SLI	10, 20, **30**, 40, 50	LEV	1, 2, **3**, 4, 5

Table 1 shows the variation of the parameters (with default values in bold). The number of attributes (ATT) allows for the evaluation of the algorithm behavior according to different data dimensionality. The number of sequences (NSQ) controls how the number of tuples per instant (equal to NSQ $\times$ 0.75) affects the algorithms. The temporal range (RAN) delimits the maximum length of the sequences and the slide interval (SLI) is related to the number of deletions when the sliding window moves.

The number of rules (RUL) and the maximum preference level (LEV) are employed in the generation of the preferences. These parameters allow us to evaluate how different preferences affect the algorithms. We used rules in the form φ_i : $\mathbf{First} \wedge Q(A_3) \rightarrow Q^+(A_2) \succ Q^-(A_2)[A_4, A_5]$ and φ_{i+1} : $\mathbf{Prev}Q(A_3) \wedge \mathbf{SomePrev}Q(A_4) \wedge \mathbf{AllPrev}Q(A_5) \wedge Q(A_3) \rightarrow Q^+(A_2) \succ Q^-(A_2)[A_4, A_5]$ having variations on propositions $Q^+(A_2)$, $Q^-(A_2)$, $Q(A_3)$, $Q(A_4)$, $Q(A_5)$. The number of iterations is RAN plus the maximum slide interval and the sequence identifier is the attribute A_1. Moreover, we executed experiments varying the number of top-k sequences (TOP). For this parameter, we used the values 4, 8, 12, 16 and 20 (8 is the default value). Greater values for TOP parameter causes more iterations in the loop of the *GetTopkSeq* algorithm.

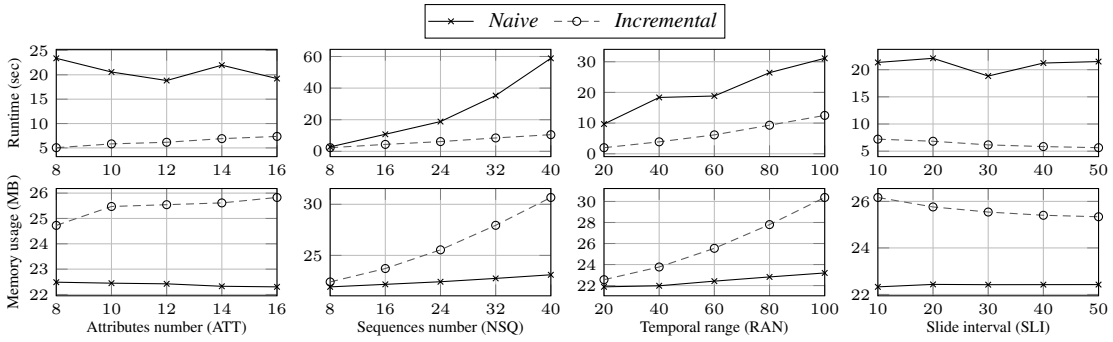


Figure 4. Experimental results for the parameters ATT, NSQ, RAN and SLI

Figure 4 shows the results obtained for the experiments with the parameters ATT, NSQ, RAN and SLI. Analyzing these results we observe that the incremental algorithm presented a better performance and a greater memory usage. This behavior is due to the maintenance of the index structure which speeds up the processing but consumes more memory. Considering the results of the experiments with the parameters NSQ and RAN,

it is possible to see that for a greater number of sequences, the algorithm has to perform more comparisons consuming more process time. Moreover, the memory usage of the incremental algorithm increases to keep an index for more sequences. It is also important to notice that the executions with bigger temporal range imply in longer sequences resulting in higher runtime and memory usage.

Figure 5 presents the results of the experiments with the parameters RUL, LEV and TOP. The results obtained are similar to the ones obtained for the other parameters. Analyzing these results it is possible to see that the incremental algorithm showed a better performance and a higher memory usage. Among these results, it is important to highlight the results obtained with the variation in the number of rules (RUL). The naive algorithm presented a poor efficiency when dealing with more rules as more comparisons are required. The incremental algorithm, however, is few affected due to its index structure as addressed in Section 4.

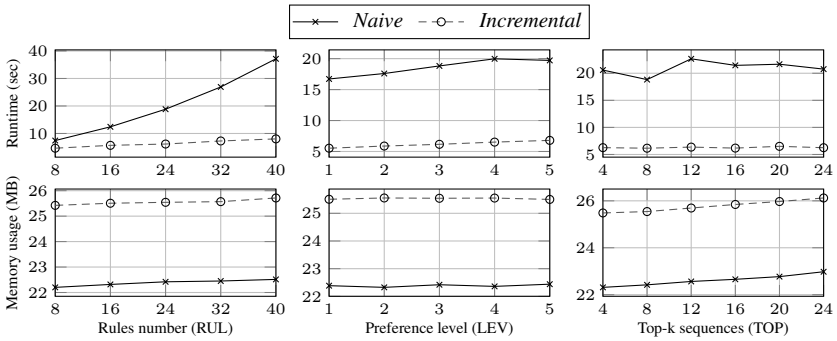


Figure 5. Experimental results for the parameters RUL, LEV and TOP

6. Related Work

The pioneering work on preference queries was the proposal of the *skyline* queries [Börzsönyi et al. 2001]. This research work provided support for many related studies. In [Chan et al. 2006], the authors introduced the concept of k-dominance. A tuple t k-dominates a tuple t' if t is better than t' in at least k attributes. The research work described in [Yiu and Mamoulis 2007] specified how to rank tuples using a *dominance degree*. The dominance degree of tuple t is the number of tuples dominated by t. The CPrefSQL query language proposed a new preference operator to compare tuples according to conditional preferences [de Amo and Ribeiro 2009].

The first research works about continuous preference queries were proposed by [Lin et al. 2005] and [Tao and Papadias 2006]. In [Lin et al. 2005] the authors explored the n-of-N problem for skyline queries, where a query is evaluated over the n most recent tuples, with $n \leq N$. The work of [Tao and Papadias 2006] designed algorithms to incrementally compute the preferred tuples over a sliding window with the most recent tuples. The work of [Kontaki et al. 2012] proposed algorithms for the evaluation of continuous preference queries over the most recent data, where each tuple has a timestamp and a validity interval. The evaluation of continuous preference queries using a graph-based index was introduced by [Santoso and Chiu 2014]. This work also designed an algorithm that outperforms the algorithms proposed in [Kontaki et al. 2012].

The first research work concerning the evaluation of continuous queries with conditional preferences (continuous cp-queries) were proposed by [de Amo and Bueno 2011, Petit et al. 2012]. In [de Amo and Bueno 2011], the authors presented an incremental algorithm based on ancestor lists for evaluating continuous cp-queries. The study described in [Petit et al. 2012] uses a graph structure to perform the same task.

In the work of [de Amo and Giacometti 2007], the authors proposed the TPref formalism to express temporal conditional preferences. The StreamPref formalism, proposed in [Ribeiro et al. 2017a], is a refinement of the TPref formalism. The StreamPref is more suitable for reasoning over data streams. In [Ribeiro et al. 2017b], the StreamPref formalism was used to define the query language *StreamPref*. The StreamPref language was originally composed of the operators **SEQ** and **BESTSEQ**. The queries using the **BESTSEQ** operator are similar to skyline queries since both return the dominant elements according to the preferences. On the other hand, the queries with **TOPKSEQ** operator are a kind of top-k dominant query. In this case, the sequences are ranked using the preference level. We also should mention the importance of the CQL language [Arasu et al. 2006]. The CQL was not designed to work with preference queries, but it is a solid and expressive SQL-based declarative language for general purpose queries over data streams. In addition, the StreamPref query language is an extension of the CQL.

7. Conclusion

This paper presented the new operator **TOPKSEQ** for the StreamPref query language. The **TOPKSEQ** uses the preference level imposed by temporal conditional preferences to find the top-k sequences. First, we revisited the existing operators **SEQ** and **BESTSEQ** of the StreamPref. The **SEQ** operator extracts sequences from a data stream and the **BESTSEQ** is used to select the dominant sequences according to temporal conditional preferences. Considering that the **BESTSEQ** operator is not enough to obtain a good result to the user in all situations, the **TOPKSEQ** can be used to complement the results obtained using sequences with higher preference level.

We demonstrated the equivalence between the operators **TOPKSEQ** and **BESTSEQ**. Moreover, we proposed an algorithm to evaluate the **TOPKSEQ** operator. This algorithm can use both the naive and the incremental strategies already proposed for the **BESTSEQ** operator. The extensive set of experiments performed showed a slightly greater memory usage for the incremental strategy recompensed by its superior performance.

Our future research directions include the possibility to use new approaches to rank the sequences beyond the preference level. We are also interested in exploring a new preference formalism to compare sequences considering not only the first different position but using a kind of distance based on preferences. Another future work is the development of algorithms for preference mining. The discovered preferences can be used in queries to monitor data streams.

Acknowledgments. The authors would like to thank the Research Agencies CNPq, CAPES and FAPEMIG for supporting this work.

References

Arasu, A., Babcock, B., Babu, S., Cieslewicz, J., Datar, M., Ito, K., Motwani, R., Srivastava, U., and Widom, J. (2016). *STREAM: The Stanford Data Stream Management System*, pages 317–336. Springer, Berlin, Heidelberg.

Arasu, A., Babu, S., and Widom, J. (2006). The CQL continuous query language: semantic foundations and query execution. *The VLDB Journal*, 15(2):121–142.

Börzsönyi, S., Kossmann, D., and Stocker, K. (2001). The skyline operator. In *ICDE*, pages 421–430, Heidelberg, Germany.

Chan, C.-Y., Jagadish, H. V., Tan, K.-L., Tung, A. K. H., and Zhang, Z. (2006). Finding k-dominant skylines in high dimensional space. In *ACM SIGMOD International Conference on Management of Data*, pages 503–514, Chicago, USA.

de Amo, S. and Bueno, M. L. P. (2011). Continuous processing of conditional preference queries. In *SBBD*, Florianópolis, Brasil.

de Amo, S. and Giacometti, A. (2007). Temporal conditional preferences over sequences of objects. In *ICTAI*, pages 246–253, Patras, Greece.

de Amo, S. and Ribeiro, M. R. (2009). CPref-SQL: A query language supporting conditional preferences. In *ACM SAC*, pages 1573–1577, Honolulu, Hawaii, USA.

Kontaki, M., Papadopoulos, A. N., and Manolopoulos, Y. (2012). Continuous top-k dominating queries. *IEEE Trans. on Knowledge and Data Eng. (TKDE)*, 24(5):840–853.

Lin, X., Yuan, Y., Wang, W., and Lu, H. (2005). Stabbing the sky: Efficient skyline computation over sliding windows. In *ICDE*, pages 502–513, Tokyo, Japan.

Petit, L., de Amo, S., Roncancio, C., and Labbé, C. (2012). Top-k context-aware queries on streams. In *DEXA*, pages 397–411, Vienna, Austria.

Ribeiro, M. R., Barioni, M. C. N., de Amo, S., Roncancio, C., and Labbé, C. (2017a). Reasoning with temporal preferences over data streams. In *FLAIRS*, Marco Island, USA.

Ribeiro, M. R., Barioni, M. C. N., de Amo, S., Roncancio, C., and Labbé, C. (2017b). Temporal conditional preference queries on streams. In *International Conference on Database and Expert Systems Applications (DEXA)*, Lyon, France.

Ribeiro, M. R., Pereira, F. S. F., and Dias, V. V. S. (2016). Efficient algorithms for processing preference queries. In *ACM SAC*, pages 972–979, Pisa, Italy.

Santoso, B. J. and Chiu, G.-M. (2014). Close dominance graph: An efficient framework for answering continuous top-dominating queries. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 26(8):1853–1865.

Tao, Y. and Papadias, D. (2006). Maintaining sliding window skylines on data streams. *IEEE TKDE*, 18(3):377–391.

Yiu, M. L. and Mamoulis, N. (2007). Efficient processing of top-k dominating queries on multi-dimensional data. In *International Conference on Very Large Data Bases (VLDB)*, pages 483–494, Vienna, Austria.

Constellation Queries over Big Data

Fabio Porto[1], Amir Khatibi[2], João N. Rittmeyer[1], Eduardo Ogasawara[3], Patrick Valduriez[4], Dennis Shasha[5]

[1]LNCC – Petropolis, RJ, Brazil

[2]UFMG – Minas Gerais, Brazil

[3]CEFET-RJ – Rio de janeiro, RJ, Brazil

[4]INRIA, Zenith, Montpellier, France

[5]NYU, Computer Science Department, New York, USA

{fporto,joaonr}@lncc.br, amir.khatibi.m@gmail.com,eogasawara@ieee.org

patrick.valduriez@inria.fr, shasha@courant.nyu.edu

Abstract. *A geometrical pattern is a set of points with all pairwise distances (or, more generally, relative distances) specified. Finding matches to such patterns has applications to spatial data in seismic, astronomical, and transportation contexts. Finding geometric patterns is a challenging problem as the potential number of sets of elements that compose shapes is exponentially large in the size of the dataset and the pattern. In this paper, we propose algorithms to find patterns in large data applications. Our methods combine quadtrees, matrix multiplication, and bucket join processing to discover sets of points that match a geometric pattern within some additive factor on the pairwise distances. Our distributed experiments show that the choice of composition algorithm (matrix multiplication or nested loops) depends on the freedom introduced in the query geometry through the distance additive factor. Three clearly identified blocks of threshold values guide the choice of the best composition algorithm.*

Resumo. *Um padrão geométrico é definido por um conjunto de pontos e todos os pares de distâncias entre estes pontos. Encontrar casamentos de padrões geométricos em datasets tem aplicações na astronomia, na pesquisa sísmica e no desenho de áreas urbanas. A solução do problema impõe um grande desafio, considerando-se o número exponencial de candidatos, potencialmente função do número de elementos no dataset e número de pontos na forma geométrica. O método aqui apresentado inclui: quadtrees,multiplicação de matrizes e junções espaciais para encontrar conjuntos de pontos que se aproximem do padrão fornecido, com um erro admissível. Apresentamos uma implementação distribuída reveladora de que a escolha do algoritmo (multiplicação de matrizes ou junções espaciais) depende da liberdade introduzida por um fator de erro aditivo na geometria do padrão. Identificamos três regiões baseadas nos valores de erro tolerados que determinam a escolha do algoritmo.*

1. Introduction

The availability of large datasets in science, web and mobile applications enables new interpretations of natural phenomena and human behavior. Consider the following use

case: **Scenario 1.** An astronomy catalog is a table holding billions of sky objects from a region of the sky, captured by telescopes. An astronomer may be interested in identifying the effects of *gravitational lensing* in quasars, as predicted by Einstein's General Theory of Relativity [Einstein 2015]. According to this theory, massive objects like galaxies bend light rays that travel near them just as a glass lens does. Due to this phenomenon, an earth telescope would receive two or more virtual images of the lensed quasar leading to a composed new object (Figure 1), such as the Einstein cross [Overbye 2015].

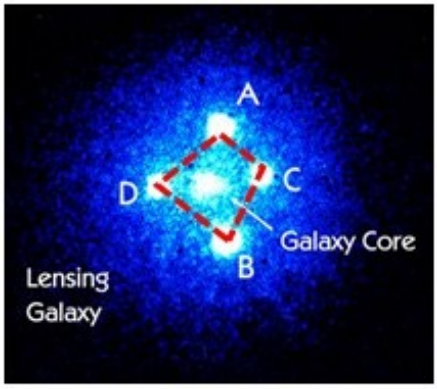


Figure 1. Einstein Cross identification from astronomic catalogs

In the scenario above, constellations, such as the Einstein cross, are obtained from compositions of individual elements in large datasets in some spatial arrangement with respect to one another. Thus, extracting constellations from large datasets entails matching geometric pattern queries against sets of individual data observations, such that each set obeys the geometric constraints expressed by the pattern query.

Solving a constellation query in a big dataset is hard due to the sheer number of possible compositions from billions of observations. In general, for a big dataset D and a number k of elements in the pattern, an upper bound for candidate combinations $\binom{|D|}{k}$ is the number of ways to choose k items from D. This paper focuses primarily on *pure constellation queries* (when all pairwise distances are specified up to an additive factor). We develop parallel algorithms that reduce the number of possible candidate sets by applying local and global constraints.

The remainder of this paper is organized as follows. Section 2 formalizes the constellation query problem. Section 3 presents our techniques to process constellation queries. In section 4, we preset our algorithms. Section 5 discusses our experimental environment and discusses the evaluation results, followed by section 6 that discusses related work. Finally, section 7 concludes.

2. Problem Formulation

In this section, we introduce the problem of answering pure Constellation Queries on a dataset of objects. A Dataset D defined as a set of elements (or objects) $D = \{e_1, e_2, \ldots, e_n\}$, in which each e_i, $1 \leq i \leq n$, is an element of a domain Dom. Furthermore, $e_i =< atr_1, atr_2, \ldots, atr_m >$, such that atr_j $(1 \leq j \leq m)$ is a value describing a characteristic of e_i.

A constellation query $Q_k = \{q_1, q_2, \ldots, q_k\}$ is (i) a sequence of k elements of domain Dom, (ii) the distances between the centroids of each pair of query elements that define the query shape and size with an additive allowable factor ϵ, and (iii) an element-wise function $f(e, q)$ that computes the similarity (e.g. in brightness at a certain wavelength) between elements e and q up to a threshold θ.

A sequence s of elements of length k in D *property matches* query Q if every element $s[i]$ in s satisfies $fe(s[i], q_i)$ up to a threshold θ and for every $i, j \leq k$: (i) the distance between elements $s[i]$ and $s[j]$ is within an additive factor ϵ of the distance between q_i and q_j, which is referred to as *distance match*. The solutions obtained using *property match* and *distance match* to solve a query Q are referred to as *pure constellations*.

3. Pure Constellation Queries

Applying *pure constellation* to find patterns such as the Einstein cross over an astronomy catalog requires efficient query processing techniques as the catalog may hold billions of sky objects.

In this context, efficiently answering pure constellation queries involves constraining the huge space of candidate sets (i.e. subsets of the catalog with the same number of stars as the query).

The next sections describe in detail the query processing techniques.

3.1. Reducing Data Complexity using a Quadtree

A constellation query looks for patterns in large datasets, such as the 2MASS catalog. Computing constellation queries involves matching each star to all neighboring stars with respect to the distances in the query, a costly procedure in large catalogs. To reduce this cost, we adopt a filtering process that eliminates space regions where solutions cannot exist.

The filtering process is implemented on top of a quadtree [Samet 1990], constructed over the entire input dataset. The quadtree splits the 2-dimensional catalog space into successively refined quadrangular regions.

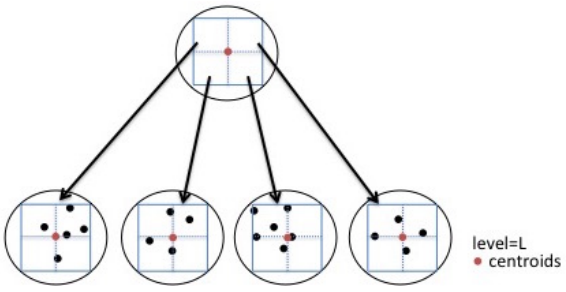


Figure 2. Quadtree node representation

A node in the tree is associated to a spatial quadrant. The geometric center of the quadrant is the node centroid and is used as a representative for all stars located in that quadrant, for initial distance matching evaluation. The quadtree data structure includes: a root node, at level $L = 0$; a list of intermediary nodes, with level $1 \leq L \leq tree\_height -$

1; and leaf nodes. To avoid excessive memory usage, data about individual stars are stored only in leaf nodes, Figure 2.

The algorithm begins by determining the level of the quadtree L_e at which the ϵ error bound exceeds the diameter of the node. If we make the reasonable assumption that ϵ is less than the minimum distance between elements in the query $(minq)$, then at height L_e no two stars would be covered by a single quadtree node.

Given a star s that will correspond to the centroid q_0 of the pattern being matched, the first step is to eliminate all parts of the quadtree that could not be relevant. The algorithm finds the node at level L_e containing s. That is called the query anchor node. The algorithm finds the nodes that lie within a radius ρ of the query anchor node, where ρ is the maximum distance plus the additive error bound ϵ between the centroid of the query pattern and any other query element. As depicted in Figure 3, in (a) a query Q has an anchor element q_0 and the largest distance to the remaining query elements $d_{0,2}$. In (b), a star is picked as an anchor and all neighboring stars within distance $d_{0,2} + \epsilon$ are preliminary candidates for distance matching.

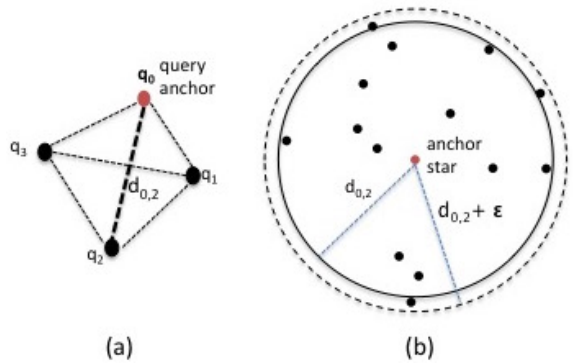


Figure 3. (a) Pure constellation query with anchor and maximum distance (b) Neighboring elements of anchor element

Next the algorithm determines for each pattern element q_i, which stars are at distance dist(q_0, q_i) from s within an additive factor of ϵ. Such stars might correspond to q_i in the case where s corresponds to q_0. For each pair of nodes n1 and n2, where n1 contains s and n2 may contain stars that correspond to q_i for some i, the algorithm checks whether the distance between the centroids of n1 and n2 matches dist(q_0, q_i), taking into account both the diameter of the nodes and the error bound. This procedure filters out all node pairs at that level for which no matching could possibly occur.

If n2 has not been filtered out, then a simple test determines whether going one level down the tree is cost effective as opposed to testing pairs of individual stars in the two nodes. That test consists of determining whether any pair of children of n1 and n2 will be eliminated from consideration based on a distance test to the centroids of those children. If so, then the algorithm goes down one level. The same matching procedure is applied to the children nodes of n1 and n2 respectively. If not, then the stars for n1 and n2 are fetched and any star in n2 that lies within an ϵ additive bound of s is put into bucket B_i.

3.2. Composition algorithms

In this section, we discuss approaches to join the buckets produced by the filtering step. As we will observe in section 5, composition algorithms are the most time consuming operation in processing constellation queries. A given anchor node may generate buckets containing thousands of elements. Thus, finding efficient composition algorithms is critical to efficient overall processing.

3.2.1. Buckets Nested Loop (Bucket-NL)

An intuitive way to produce constellations for a given anchor element is by directly joining the buckets of candidate elements considering the corresponding pairwise distances between query elements as the join predicate. In this approach, each bucket is viewed as a relation, having as a schema their spatial coordinates and an id, $B_i(starid, ra, dec)$. A solution is obtained whenever a tuple is produced having one neighbor element from each bucket, such that the distances between each element in the solution *distance-match* those among respective query elements, +/- ϵ. Bucket-NL assumes a nested loop algorithm to traverse the buckets of candidate elements and checks for the distance predicates. Thus, applying a *distance-match* constraint corresponds to applying a cyclic join among all buckets in the bucket set followed by a filter among non-neighbors in the cycle. For example, Bucket-NL would find pairs (t1, t2) where t1 is from B_i with and t2 from B_{i+1} if dist(t1, t2) is within dist(p_i, p_{i+1}) +/- ϵ. Then given these pairs for buckets 1 and 2, buckets 2 and 3, buckets 3 and 4, etc, Bucket-NL will join these cyclically and then for any k-tuple of stars s1, s2, ..., sk that survive the join, Bucket-NL will also check the distances of non-neighboring stars (e.g. check that dist(s2,s5) = dist(p_2, p_5) +/- ϵ).

3.2.2. Matrix Multiplication based approaches

The Matrix Multiplication (MM) based approaches precede the basic $Bucket\_NL$ algorithm by filtering out candidate elements. Here are the details: recall that bucket B_i holds elements for the candidate anchor that correspond to dist(q_0, q_i) +/- ϵ. Compute the matrices: $M1(B_1, B_2), M2(B_2, B_3), M3(B_3, B_1)$ where $Mi(B_i, B_{i+1})$ has a 1 in location j, k if the jth star in B_i and the kth star in B_{i+1} is within dist(p_i, p_{i+1}) +/- ϵ. The product of matrices indicates the possible existence of solutions for a given anchor element, as long as the resulting matrix contains at least a one in its diagonal. The MM approach can be implemented with fast matrix multiplication algorithms [R. Bank 1993][U. Zwick 2005] and enables quick elimination of unproductive bucket elements.

3.2.3. MMM Filtering

Matrix multiplication may be applied multiple times to eliminate stars that cannot be part of any join. The idea is to apply k matrix multiplications, each with a sequence of matrices starting with a different matrix (i.e. a B_i bucket appears in the first and last matrices of a sequence, for $1 \leq i \leq k$). The resulting matrix diagonal cells having zeros indicate that the corresponding element is not part of any solution and can be eliminated. For example, for buckets B_1, B_2, B_3 and matrices $M1(B_1, B_2), M2(B_2, B_3), M3(B_3, B_1)$, we would

run $< M1 \cdot M2 \cdot M3 >$; $< M2 \cdot M3 \cdot M1 >$ and $< M3 \cdot M1 \cdot M2 >$. For the multiplication starting with say $M1$, elements in bucket B_1 with zeros in the resulting matrix diagonal are deleted from B_1, reducing the size of the full join.

3.2.4. Matrix Multiplication Compositions

The matrix multiplication filtering is coupled with a composition algorithm leading to $MM\_Composition$ algorithms. The choices explore the tradeoff between filtering more by applying the *MMM* filtering strategy or not.

The *MMM_NL* strategy uses the *MMM* filtering strategy to identify the elements of each bucket that do not contribute to any solutions and can be eliminated from their respective buckets. Next, the strategy applies *Bucket_NL* to join the buckets with elements that do contribute to solutions.

The *MM_NL* considers a single bucket ordering with the anchor node bucket at the head of the list. Thus, once the multiplication has been applied, elements in the anchor node bucket that appear with zero in the resulting matrix diagonal are filtered out from its bucket. Next, the strategy applies *Bucket_NL* to join the buckets with anchor elements that do contribute to solutions.

4. Algorithms for Pure Constellation Queries

To compute *Pure Constellation Queries*, the overall algorithm implements *property matching* and finds matching pairs, whereas the composition algorithms implement *distance matching* as discussed above.

4.1. Main Algorithm

The Constellation Algorithm depicts the essential steps needed to process a Constellation query. The main function is called *ExecuteQuery*. It receives as input a query q, dataset D, element predicate fe, similarity threshold θ, and error bound ϵ. At step 1, a quadtree entry level L_e is computed. Next, a quadtree qt is built covering all elements in D and having height L_e. Figure 2.a illustrates a typical quadtree built on top of heterogeneously distributed spatial data. The quadtree nodes at level L_e become the representatives of stars for initial distance matching. Considering the list of nodes at level L_e, an iteration picks each node, takes it as an anchor node, and searches qt to find neighbors. The geometric centroid of the node quadrant is used as a reference to the node position and neighborhood computation. Next, each pair (anchor node, neighbor) is evaluated for distance matching against one of the query pairs: (query anchor, query element) and additive factor ϵ. Matching nodes contribute with stars for distance matching or can be further split to eliminate non-matching children nodes. Matching stars are placed in a bucket holding matches for the corresponding query element.

The *Compose Function*, applies a composition algorithm, described in the previous section, between buckets $B = \{B_1, B_2, \ldots, B_k\}$, for $q.size = k + 1$, to see which k+1-tuples match the pure constellation query. The composition algorithm builds a query execution plan to join buckets in B. The distance matching of elements in buckets B_i and B_j, $i \neq j$, and $i, j \neq anchor$, is applied by checking their pairwise distances +/- ϵ, with respect to the corresponding distances between q_i and q_j, in q.

The choice between running *Bucket_NL* or a *MM_filtering* algorithm to implement element composition, as our experiments in section 5 will show, is related to the size of the partial join buckets. For dense datasets and queries with an error bound close to the average distance among stars, lots of candidate pairs are produced and *MM_filtering* improves composition performance, see Figure 4.b.

5. Experimental Evaluation

In this section, we start by presenting our experimental setup. Next, we assess the different components of our implementation for Constellation Queries.

5.1. Set Up

5.1.1. Dataset Configuration

The experiments focus on the Einstein cross constellation query and are based on an astronomy catalog dataset obtained from the Sloan Digital Sky Survey (SDSS), a seismic dataset, as well as synthetic datasets. The SDSS catalog, published as part of the data release DR12, was downloaded from the project website link ($http : //skyserver.sdss.org/CasJobs/$).
We consider a projection of the dataset including attributes $(objID, ra, dec, u, g, r, i, z)$. The extracted dataset has a size of 800 MB containing around 6.7 million sky objects. The submitted query to obtain this dataset follows:

Select objID, ra, dec, u, g, r, i, z
From PhotoObjAll into MyTable

From the downloaded dataset, some subsets were extracted to produce datasets of different size. Additionally, in order to simulate very dense regions of the sky, we built synthetic datasets with: 1000, 5000, 10000, 15000, and 20000 stars. The synthetic dataset includes millions of scaled solutions in a very dense region. Each solution is a multiplicative factor from a base query solution chosen uniformly within an interval of scale factors $s = [1.00000001, 1.0000009]$.

5.1.2. Calibration

We calibrated constellation query techniques using the SDSS dataset described above and a 3D seismic dataset from a region on the North Sea: Netherlands Offshore F3 Block Complete [1]. The procedure aimed at finding the *Einstein Cross* in the astronomy catalog and a seismic dome within the North Sea dataset, using our constellation query answering techniques. In both cases, the techniques succeeded in spotting the right structures among billions of candidates.

5.1.3. Computing Environment

The Constellation Query processing is implemented as an Apache Spark dataflow running on a shared nothing cluster. The Petrus.lncc.br cluster is composed of 6 DELL PE R530

[1] https://opendtect.org/osr/pmwiki.php

servers running CENTOS v. 7.2, kernel version $3.10.0327.13.1$ $.el7.x86\_64$. Each cluster node includes a 2 Intel Xeon E5-2630 V3 2.4GHz processors, with 8 cores each, 96 GB of RAM memory, 20MB cache and 2 TB of hard disk. We are running Hadoop/HDFS v2.7.3, Spark v2.0.0 and Python v2.6. Spark was configured with 50 executors each running with 5GB of RAM memory and 1 core. The driver module was configured with 80GB of RAM memory. The implementation builds the quadtree at the master node, at the driver module, and distributes the list of nodes at the tree entry level. Each worker node then runs the *property_matching* and *distance_matching* algorithms. Finally, answers are collected in a single solution file.

5.2. The Effectiveness of the Descent Tree algorithm

The quadtree structure enables reducing the cost of constellation query processing by restricting composition computation to stars in pairs of nodes whose spatial quadrants match in distance. Selected matching pair nodes are evaluated for further splitting, according to cost model. In this section, we investigate the efficiency of the algorithm. We compare the cost of evaluating the stars matching at the tree entry level with one that descends based on the cost model.

We ran the *buildQuadtree* function with dense datasets and measured the difference in elapsed-time in both scenarios. In terms of number of comparisons for 1 million stars, the cost model saves approximately 1.9x, leading to an order of magnitude on execution time savings.

5.3. Composition Algorithm Selection

In this section, we discuss the characteristics of the proposed composition algorithms.

In the first experiment a constellation query based on the Einstein cross elements is run for each composition algorithm and their elapsed-times are compared. The elapsed-time values correspond to the average of 10 runs measuring the maximum among all parallel execution nodes in each run.

The geometric nature of constellation queries and the density of astronomical catalogs make the distance additive factor ϵ a very important element in query definition. As our experiments have shown, variations in this parameter may change a null result set to one with million of solutions. The experiments evaluate two classes of composition algorithms. In one class, we use the *Bucket_NL* algorithm and, in the second one, we include the adoption of various *Matrix Multiplication* filtering strategies.

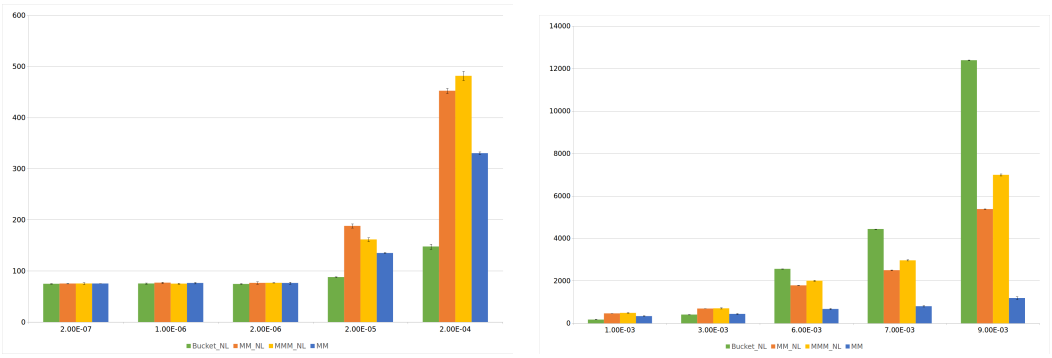
The experiment results are depicted in Figures 4.a and 4.b. In these plots, the horizontal axis presents different error tolerance values ϵ, while the vertical axis shows the elapsed-time of solving the constellation query using one of the composition algorithms.

Figure 4.a shows basically two scenarios. For very small ϵ, $\leq 10^{-6}$, the number of candidate elements in buckets is close to zero, leading to a total of 32 anchor elements to be selected and producing 52 candidate shapes. In this scenario, the choice of a composition algorithms is irrelevant, with a difference in elapsed-time of less than 10% among them. It is important to observe, however, that such a very restrictive constraint may eliminate interesting sets of stars. Unless the user is quite certain about the actual shape of its constellation, it is better to loosen the constraint.

The last blocks of runs involving the composition algorithms in Figure 4.a shows that the results are different when increasing ϵ by up to a factor of 100. Considering $\epsilon = 2,0 \times 10^{-5}$,we obtain $522,578$ productive anchor elements and an average of close to one element per bucket. The total number of candidate shapes rises to 12.6 million. In this setting, $Bucket\_NL$ is very fast, as it loops over very few elements in the buckets to discover solutions. The overhead of computing matrix multiplication is high, so $Bucket\_NL$ is a clear winner. This scenario continues to hold up to $\epsilon = 2 \times 10^{-4}$, see Figure 4.a. In this range, $Bucket\_NL$ is faster than MM_NL and MMM_NL by 214% and 240%, respectively.

Figure 4.b highlights the behavior of algorithms under additive error tolerance values. The flexibility introduced by $\epsilon = 6 \times 10^{-3}$ generates 6.7 million productive anchor elements and a total of 7.1 billion solutions, with average elements per bucket of 10. In this scenario, eliminating non-productive anchor elements, close to $300,000$, by filtering using matrix multiplication eliminates the need of computing nested loops over approximately 405 candidate elements in buckets. Thus, running fast matrix multiplication as a pre-step to nested-loop becomes beneficial.

Figure 5 shows the point at which matrix multiplication becomes beneficial: when $\epsilon = 6.0 \times 10^{-3}$ matrix multiplication starts to efficiently filter out anchor nodes and so the reduction in nested-loop time compensates for the cost of performing matrix multiplication. The gains observed by running matrix multiplication algorithms as a pre-filtering step before nested loop for ϵ in range 6×10^{-3} and 9×10^{-3} are up to 45.6% for *MM_NL* and 34.6% for *MMM_NL*.



(a) (b)

Figure 4. Low Additive Error Tolerance ϵ (a) High Additive Error Tolerance ϵ (b)

Figures 5.a and 5.b zoom in on the *Matrix Multiplication* algorithms. The former shows the results with thresholds not less than 1×10^{-3}. In this range, we can observe an inversion in performance between MM_NL and MMM_NL. The inflexion point occurs after $\epsilon \geq 2 \times 10^{-3}$. Threshold values below the inflexion point include anchor nodes with very few elements in buckets. In this scenario, computing multiple matrix multiplication is very fast. Moreover, elements that appear with zeros in the resulting matrix diagonal can be looked up in buckets and deleted, before the final nested-loop. The result is a gain of up to 14% in elapsed-time with respect to MM_NL. From the inflexion point on, $Matrix\_Multiplication\_NL$ is the best choice with gains up to 30% with respect to MMM_NL. The selection among composition algorithms is summarized in Table 1,

according to the results on the SDSS dataset.

Table 1. Composition Algorithms Selection

Threshold-Range	Best Choice Composition Algorithm
≤ 0.003	$Bucket\_NL$
> 0.003	MM_NL

Finally, the matrix multiplication *MM* algorithm is, as expected, a good choice for existential constellation queries which ask whether any subset of the dataset matches the query but does not ask to specify that subset. In this scenario, once the matrix multiplication indicates a resulting matrix diagonal with all zeros, the anchor element produces no candidate shape and can be eliminated from the existential query result.

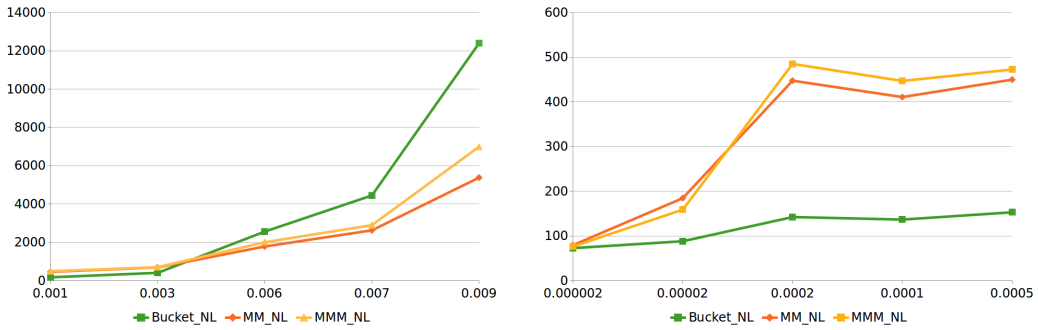


Figure 5. Zoom In on Matrix Multiplication: large threshold (a) Zoom In on Matrix Multiplication: low threshold (b)

5.4. Pure Constellation Scale-up

We investigated Pure CQ scale-up adopting the set of dense datasets (see section 5.1.1), error bound epsilon = $4.4x10^{-6}$ and Bucket_NL for the composition algorithm. The execution produced solutions of size: zero, 21, 221, 1015, and 2685. The run with 1000 stars dataset produced zero solutions, which shows the relevance of tunning the error bound for a given dataset and the restrictions imposed by Pure CQ. Apart from the runs with the 15,000 stars dataset, the variations in time followed the increase in the number of solutions.This indicates that non solutions are quickly discarded and the time is mostly due to producing solutions. Figure 6 depicts the results, where time corresponds to the elapsed-time in seconds of the parallel execution.

6. Related Work

Finding collections of objects having some metric relationship of interest is an area with many applications. The problem has different names depending on the discipline, including *Object Identification* [Singla and Domingos 2005], *Graph Queries* [Zou et al. 2011], *Pattern Matching* and *Pattern Recognition* [Bishop 2006].

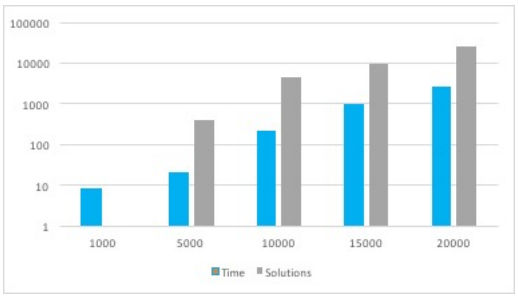


Figure 6. Pure Constellation Scale-up

Pattern recognition research focuses on identifying patterns and regularities in data [Bishop 2006]. Graphs are commonly used in pattern recognition due to their flexibility in representing structural geometric and relational descriptions for concepts, such as pixels, predicates, and objects [Jolion 2001]. In this way, problems are commonly posed as a graph query problem, such as subgraph search, shortest-path query, reachability verification, and pattern match. Among these, subgraph matching queries are related to our work.

In a subgraph query, a query is a connected set of nodes and edges (which may or may not be labeled). A match is a (usually non-induced) subgraph of a large graph that is isomorphic to the query. While the literature in that field is vast [[Zou et al. 2009], [Giugno and Shasha 2002]], the problem is fundamentally different, because there is no notion of space (so data structures like quadtrees are useless) and there is no distance notion of scale (the ϵ that plays such a big role for us).

Finally, constellation queries are a class of package queries (PQ), Brucato et al. [Brucato et al. 2016].

7. Conclusion

In this paper, we introduce *constellation queries*, specified as a geometrical composition of individual elements from a big dataset. We illustrate the application of Constellation Queries in astronomy (e.g. Einstein crosses).

We have designed procedures to efficiently compute both pure Constellation Queries. First, we reduce the space of possible candidate sets by associating to each element in the dataset neighbors at a maximum distance, corresponding to the largest distance between any two elements in the query. Next, we filtered candidates yet further into buckets through the use of a quadtree. Next, we used a bucket joining algorithm, optionally preceded by a matrix multiplication filter to find solutions.

Our experiments execute on Spark, running on the neighboring dataset distributed over HDFS. Our work shows that our filtering techniques having to do with quadtrees are enormously beneficial, whereas matrix multiplication is beneficial only in high density settings.

There are numerous opportunities for future work, especially in optimization for higher dimensions.

8. Acknowledgment

This research is partially funded by EU H2020 Program and MCTI/RNP-Brazil(HPC4e Project - grant agreement number 689772), FAPERJ (MUSIC Project E36-2013) and INRIA (SciDISC 2017), INRIA international chair, U.S. National Science Foundation MCB-1158273, IOS-1139362 and MCB-1412232. This support is greatly appreciated.

References

Bishop, C. M. (2006). Pattern recognition and machine learning. *Springer*, page 7.

Brucato, M., Beltran, J. F., Abouzied, A., and Meliou, A. (2016). Scalable package queries in relational database systems. *Proc. VLDB Endow.*, 9(7):576–587. 00004.

Einstein, A. (2015). Relativity: The special and the general theory.

Giugno, R. and Shasha, D. (2002). GraphGrep: A fast and universal method for querying graphs. In *Proceedings - International Conference on Pattern Recognition*, volume 16, pages 112–115. 2 edition. 00125.

Jolion, J. (2001). Graph matching : what are we really talking about. *Proceedings of the 3rd IAPR Workshop on Graph-Based Representations in Pattern Recognition*.

Overbye, D. (2015). Astronomers observe supernova and find they're watching reruns. New York Times, USA.

R. Bank, C. D. (1993). Sparse matrix multiplication package (smmp). *Advances in Computational Mathematics*, 1:127–137.

Samet, H. (1990). *The Design and Analysis of Spatial Data Structures*. Addison-Wesley.

Singla, P. and Domingos, P. (2005). Object identification with attribute-mediated dependences. *Proceedings of the 9th European conference on Principles and Practice of Knowledge Discovery in Databases*.

U. Zwick, R. Y. (2005). Fast sparse matrix multiplication. *ACM Transactions on Algorithms (TALG)*, 1:2–13.

Zou, L., Chen, L., and Özsu, M. T. (2009). Distance-join: Pattern Match Query in a Large Graph Database. *Proc. VLDB Endow.*, 2(1):886–897.

Zou, L., Chen, L., Özsu, M. T., and Zhao, D. (2011). Answering pattern match queries in large graph databases via graph embedding. *The VLDB Journal*, 21:97–120.

Emotion analysis of reaction to Terrorism on Twitter

Jonathas G. D. Harb[1], **Karin Becker**[1]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

jonathasgabriel05@gmail.com, karin.becker@inf.ufrgs.br

Abstract. *Terrorism events impact people in several manners. Reactions may include losing sense of safety and experiencing angry and fear, among others. The social media has become an important mean where people express themselves. We target Twitter to investigate the emotional reaction people have to terrorism events. For this purpose, we analyze emotions in tweets along with demographic data. Tracking emotional reaction can help in defining specific assistance programs. In our approach we collect a corpus of tweets related to two terrorism events, classify emotions, extract user location and estimate user age and gender with use of available tools. Results showed an emotion shift due to the events and a difference on the reaction from one event to another.*

1. Introduction

Terrorism in all its forms remains a constant threat that continues to be present in the global agenda and raises questions concerning prevention and consequences. In general, terrorism involves the use, or threat of use, of violence as an attempt to achieve some social or political effect. The goal of terrorism is to create instability by propagating fear, arousal and uncertainty on a wider scale than those achieved by targeting a single victim (Horgan 2014). Terrorism attempts are becoming more frequent and diverse, and reactions to terrorism events include, among others, loosing sense of safety, feeling helpless, experiencing anger and fear. Tracking user emotions can help authorities to define and provide specific assistance programs for coping with it.

Today's widely accessible social micro-blogging platforms such as Twitter are increasingly being used on global scale to publish content and express emotions and opinions on a daily basis. This large volume of information are being explored by data science area for several purposes, such as to identify sentiment and emotions expressed in tweets (Mohammad et al. 2015; Mohammad 2012), to monitor how people feel about specific topics (Wan and Paris 2015), to predict information flow size and survival following specific events (Burnap et al. 2014), to analyze social connections (Lerman et al. 2016), and to study engagement to context-specific tweets (Suttles and Ide 2013), among others.

One opportunity is to explore such information to investigate users' emotional reaction to terrorism events, through emotion analysis. Sentiment Analysis is the field of study that analyzes people's opinions, appraisals, evaluations, attitudes, and emotions from written language (Liu 2012). Emotion mining involves identifying emotion bearing words/expressions in texts and classifying them according to an emotion model (Munezero et al. 2014). Common approaches for sentiment classification are the adoption of emotion lexicons and supervised learning over emotion-labeled data

resources. These resources are less abundant, when compared to polarity classification. Popular emotion models are basic emotions and VAD (valence, arousal and dominance) (Munezero et al. 2014).

Sentiment analysis in tweets is difficult due to its unstructured and informal short text. By design, users have a limited number of 140 characters, and typically tweets contain casual text with errors (spelling, grammar, etc), abbreviations, internet-specific terms, etc. In addition, working with emotions in tweets is a much less studied problem by the literature due to the lack of labelled data (Hasan et al. 2014).

Previous works have focused on sentiment analysis in texts, such as tweets, with different goals. While authors in (Mohammad et al. 2015; Mohammad 2012) applied different labelling techniques, others have proposed novel approaches for classifying tweets into sentiment categories (Wang et al. 2012; Purver and Battersby 2012; Bravo-Marquez et al. 2016; Kim 2014). Works have also focused on analyzing social aspects of emotion in twitter (Kim et al. 2011) and on using demographic data to characterize social relations (Lerman et al. 2016) and population mobility patterns (Gallegos et al. 2015). Emotional reaction to specific topics or events was the focus in (Wan and Paris 2015). However, studies on sentiment analysis to investigate emotional reaction to terrorism in Twitter are still lacking.

In this paper, we aim to investigate and help understand the emotions people express about terrorism events with help of demographics data. For that purpose, we collected and analyzed data on two terrorism events that occurred in England, in order to answer the following questions:

- Q1: Is there an emotion shift due to terrorism events?
- Q2: Do different terrorism events raise the same emotional reaction?
- Q3: Do user location have an impact on the emotional reaction?

To answer these questions, we collected a corpus of terrorism-related tweets, identified the presence of emotions with deep learning methods, and determined demographic information of the respective users, such as location, age and gender. To the best of our knowledge, this is the first work that investigates emotional reaction in Twitter in the specific context of terrorism.

The remaining of this paper is structured as follows. Section 2 describes related work. Section 3 describes the methods and materials used for providing answers to our research questions. Section 4 describes our experiments to find a suitable model for emotion prediction. Section 5 presents the analysis performed over the data to answer our questions. Finally section 6 presents the conclusion and opportunities for future works.

2. Related Work

Sentiment analysis was target of several works for different purposes. A series of approaches for sentiment classification and data labelling were presented in (Mitchell et al. 2013; Anderson 2005; De Choudhury et al. 2016; Lotan et al. 2011). The work presented in (ElSherief et al. 2017) applied distant supervision and machine learning methods such as Naive Bayes (NB) and Maximum Entropy (ME) for sentiment classification. Kim (Kim 2014) applied deep learning in natural language text by training a convolutional neural networks (CNN) on top of pre trained word embeddings. Models

were evaluated against several datasets and the results outperformed several state of the art methods in the majority of the experiments.

Previous works have also explored the social questions involving tweet sentiment analysis. Lerman et al. (2016) analyzed a large corpus of georeferenced tweets in order to study the structure of social connections people form online. Their work collected tweets from US areas, linked these tweet locations to corresponding Census data and estimated tweet sentiment into negative and positive through SentiStrenght[1]. In their analysis, they were able to identify groups that expressed more positive emotions as well as groups where negative emotions were predominant. The structure of social connections of these groups as well as demographic data helped in explaining such findings.

Suttles et al. (2013) studied user engagement with Gender-based violence (GBV) related posts in Twitter. Age, gender and linguistic attributes, including emotions, were analyzed. Their work reported how users engage with GBV tweets based on favoriting and retwitting metrics. Descriptive statistical analysis was applied to identify age and gender of tweeters. Tweet sentiment was extracted from the LIWC software[2]. They found that users engage more to GBV related tweets than to generic tweets and that the engagement is not uniform across genders and ages. Moreover, anger was often predominant in the GVB content context.

Gallegos et al. (2015) used data from Foursquare service to identify US metropolitan areas that people use to check-in. These areas were then analyzed with help of demographics data. Their results revealed that areas with many check-ins have happier tweets and therefore encourage other people to connect to these places. They reported that such results provided more information on human mobility patterns.

Despite all cited contributions, just a few studies have focused on emotion mining in their tasks and none of the them have studied sentiment analysis in a specific context as we do by targeting terrorism events.

3. Materials and Methods

3.1. Dataset

We decided to target two terrorism events that occurred in the United Kingdom. This choice was motivated by two factors. First, we focused on the English language, in order to benefit from many tools and functions available for natural language processing. Second, to study emotional reaction for different events in a same region, as England was the target of a few attacks in 2017.

The first event was the Manchester Arena bombing[3], which took place on May 22th, 2017 in Manchester, when people were leaving a concert of Ariana Grande. The second one was the London Bridge attack, occurred in London on June 3rd 2017, where a van left the road and struck a number of passing by pedestrians[4].

Data collection must involve tweets from the past, as the occurrence of a terrorism event is unpredictable. As the Twitter official streaming API does not allow to collect

[1]http://sentistrength.wlv.ac.uk/

[2]http://liwc.wpengine.com

[3]https://en.wikipedia.org/wiki/Manchester_Arena_bombing

[4]https://en.wikipedia.org/wiki/June_2017_London_Bridge_attack

Table 1. Query Terms, Dates and Dataset per Event

Event Name	Query terms	Period	BEFORE (#tweets)	AFTER (#tweets)
#prayformanchester	#prayformanchester, "Manchester"	05-20-2017 to 05-24-2017	BM (5,351)	AM (25,010)
#londonbridge	#londonBridge, "London"	06-01-2017 to 06-05-2017	BL (20,379)	AL (29,656)

tweets from the past, we used an open source project[5] written in Python, which bypasses some of the limitations of Twitter API. As parameters, we set query search terms combined with boundary dates.

For each targeted event, we collected tweets two days before the event, the actual day it happened, and two days after the event. In this way, we were able to analyze not only emotion reaction, but also a possible emotion shift due to the events. To define search terms to collect tweets, we analyzed raw data gathered from the web, trending topics, as well as samples extracted using the official Twitter API on the respective dates. We found recurrent hashtags for each one of the events, namely #prayformanchester for the Manchester attack and #londonbridge for the London one. We assumed these hashtags were representative due to their major predominance in tweets referring to these events (nearly 10 times more frequent). Tweets collected two days before the events were queried by the keywords "Manchester" and "London". We considered these tweets as representative, as we observed that these keywords were commonly used to tweet about citizen's thoughts on diverse topics such as football teams, universities, and daily news regarding these cities, among others. Table 1 shows queries search terms and boundary dates for tweet collection.

Data pre-processing involved traditional steps, such as the removal of hyperlinks, hashtags (because they did not provide useful information other than identifying the events), mentions to other users, special marks and symbols (&, /, ; _, etc). In addition, we applied an English dictionary[6] to filter out tweets with too many misspelled words and non English ones. These actions resulted in four datasets (shown in table 1): BM (before Manchester) containing 5,351 tweets, BL (before London) containing 20,379 tweets, AM (after Manchester) containing 25,010 tweets, and AL (after London) containing 29,656 tweets. The structure of these datasets is identical and include, among others, the filtered tweet text and the tweet ID.

We characterized the demographics of the data in terms of location, gender and age. For location, we observed that less than 1% of the collected tweets were georeferenced. Thus, we assumed that the location of the tweet would be extracted from the users' profile as in (Sakaki et al. 2010). Our original idea of analyzing sentiment per city could not be accomplished due to the low number of tweets for comparison. On the other hand, in analyzing location by countries and larger regions, we found out that a representative number of locations from the UK and the US were present in the dataset Therefore, we filtered locations in three categories: locations from the UK, locations from the US and "other locations". As the location in each user profile is a simple text without any validation, we compared each declared location against a list of cities from the UK and the US to include in these two categories. As in (ElSherief et al. 2017), gender and age were esti-

[5]https://github.com/Jefferson-Henrique/GetOldTweets-python

[6]https://github.com/dwyl/english-words

Table 2. Gold Standard: Number of labelled tweets per category

Emotion	Anger	Disgust	Fear	Sadness	Surprise	None
# tweets	82	116	85	179	71	74

mated using Face++[7]. Face++ provides an API that analyzes face related attributes based on machine learning, and experiments evaluated an accuracy of 85% (Fan et al. 2014)

3.2. Gold Standard

Our work focuses on five out of the the six basic emotion categories defined by Eckman (Ekman and Friesen 1982). We focused on negative emotions only, because we assume people are not likely to express positive emotions (such as happiness) in reaction to terrorism events[8]. The emotion categories considered therefore include anger, fear, sadness, surprise and disgust. Our approach considers that a given tweet is included in one and only one of the emotion categories.

To train a model for emotion prediction, an emotion labeled dataset is required. As domain-related datasets tend to provide the best results (Liu 2012), we created a specific terrorism gold standard for the task. Tweets were labeled according to each emotion category considered, plus an extra "none" category. This was accomplished using Amazon Mechanical Turk[9].

First, one of the authors annotated 967 tweets with the considered 5 emotion labels, based on the presence of emotion keywords and expressions. For example, the tweet "*Deeply saddened by the loss of 22 beautiful lives. we should not live like this. They did not deserve to die*" was labelled as sadness due to the expression "deeply saddened"; the tweet "*It's so scary to not feel safe in this World*" was labelled as fear due to the expression "It's so scary", and so on. We started from a randomly selected set of tweets, discarding the ones that did not contain an unquestionable emotion word/expressions, and labeling otherwise. This task was performed until we reached a minimum of 100 tweets per emotion. This procedure resulted in relatively well balanced sets. Afterwards, we created a HIT (Human Intelligence Task) with these tweets, where annotators were asked to determine which emotion best described a tweet, given a set of categories as options (anger, fear, disgust, sadness, surprise, none). We instructed annotators to choose the primary emotion if more than one emotion could be identified, and to choose "none" if no emotion could be clearly determined. We targeted the HIT to two master annotators, so that we would have three annotators in total, considering one of the authors. According to Amazon, master annotators typically have a 90% or more of accuracy rate. We filtered out tweets in which there was a disagreement between all the three annotators, and retained those with at least two agreements. The results, composed of 607 tweets, are displayed in Table 2, which we consider as our ground truth for validating the emotion prediction model.

[7]https://www.faceplusplus.com/

[8]https://www.paulekman.com/blog/our-emotional-reactions-terrorism/

[9]https://www.mturk.com/

3.3. Classification

In order to classify our collection of tweets, we applied deep learning by training a Convolutional Neural Network (CNN) as defined in (Kim 2014). Our choice is due to their results, and the pioneering in using such approach for classifying natural language. The results of the model presented in (Kim 2014) outperformed traditional methods, such as Support Vector Machine (SVM), in a variety of text classification tasks and since then it is widely referenced in the literature. Another motivating factor was the automatic learning capability that deep learning has by incorporating improved learning procedures that make use of computing power and training data, working well on large sets of data (Ain et al. 2017). In a nutshell, the CNN architecture comprises four layers. The first layer converts words into vectors of low-dimensional representation called *word embeddings*. The second layer applies a series of convolutions over these word embeddings to produce a feature map for each sentence. The third layer is responsible for filtering the most important features into one feature vector trough a max polling operation. The fourth layer applies the softmax function to classify sentences into labels. The Python code of the CNN implementation we used is publicly available [10] [11] [12], and it is designed to be executed on the top of TensorFlow[13], an open source software library for high performance numerical computation.

4. Experiments

We developed a few experiments with our CNN in order to find the most suitable classification model for our emotion categories. The CNN parameters we used were the same as in (Kim 2014) because their results were built using these parameters, and all the variations we tried did not provide significant difference on our results. Our experiments were focused on the input provided to the CNN, which is the training set. Given that our limited number of labelled tweets did not provide enough data for properly training the CNN, we tried different approaches for gathering enough training seeds for our emotion categories:

- Distant supervision (Go et al. 2009; Purver and Battersby 2012; Suttles and Ide 2013): we applied distant supervision and used the emotion-labelled electoral tweets provided by (Mohammad et al. 2015) as training seeds. This resulted in 2,575 seeds.
- Filtering by keywords: we analyzed samples of our dataset and defined keywords that were likely to represent emotions in a tweet. The process for obtaining our keywords set was the same as for labeling tweets for our gold standard. We randomly selected sets of tweets and identified specific keywords that indicated presence of emotions of our emotion categories. Afterwards we checked other samples for such keywords and verified that tweets containing them were likely to belong to the respective emotion category, we also confirmed the presence of such keywords in our gold standard. We then filtered the tweets by these keywords and considered them as training seeds. This resulted in 4,019 seeds. Keywords used for filtering can be seen in Table 3.

[10]https://github.com/cahya-wirawan/cnn-text-classification-tf
[11]https://github.com/cahya-wirawan/cnn-text-classification-tf
[12]https://github.com/dennybritz/cnn-text-classification-tf
[13]https://www.tensorflow.org/

Table 3. Keywords used for filtering training seeds for the CNN

Emotion	Keywords
anger	anger, fuck, fucked, pissed, lmaof, damm
disgust	disgust, disgusted, disgusting
fear	worried, worry, scary, scaring, scared, fear
sadness	sad, sadness, saddened
surprise	surprised, surprising, surprise, shocked, shocking

Table 4. Results for the generated CNN prediction models

Approach	Avg. Precision	Avg. Recall	Avg. F-measure
Distant Supervision	0,4049	0,2099	0,1087
Keywords	0,7348	0,718	0,6846
Hashtags	0,322	0,3014	0,2783
Dictionary-based	0,5666	0,2958	0,2722

- Filtering by hashtags (Mohammad 2012): we used emotion hashtags collected from (Mohammad 2012) to provide automatic labelling. Labelled tweets were used as seeds. This approach resulted in only 150 seeds.
- Dictionary-based filtering: we used a lexicon approach and filtered tweets with the emotion categories available in NRC (Mohammad and Turney 2013). Tweets in which one emotion prevailed were filtered and used as training seed. This resulted in 23,153 seeds.

For each approach, the CNN was trained and a prediction model was generated. In all of our experiments, training seeds for the ”none” category were chosen by selecting tweets that did not contain any of the following terms: a) defined keywords used as seeds (Table 3), b) emotional hashtags defined in (Mohammad 2012), c) emotion expressions labeled according to the NRC lexicon (Mohammad and Turney 2013). We randomly selected 1,000 tweets for the ”none” class. The test was always conducted against our labelled set of tweets. To improve our results, we did as in (Kim 2014) and loaded in our CNN pre-trained word embeddings for all the experiments. We chose the word embeddings corpus provided by GloVe [14] because it is extracted specifically from tweets. Following (Kim 2014), the use of pre-trained word embeddings is an approach commonly used to improve performance when the training set is not large enough. Incorporating the GloVe's embedding set in the CNN improved our results. General results of our models can be found in Table 4.

As we can see, distant supervision did not provide the best of the results. One explanation could be due to the peculiarities of our context, which includes words and expressions different than those of an electoral debate context. The approaches based on lexicon and hashtags did not provide good results as well. We noticed a very high level of absence of emotion hashtags in our dataset, which resulted in very few seeds, not enough to generate an accurate prediction model.

From all of our experiments, the one filtering by keywords provided the best results and therefore was the one used to generate our prediction model. We con-

[14]https://nlp.stanford.edu/projects/glove/

Table 5. F-measure for the model generated by filtering keywords

Emotion	anger	disgust	fear	sadness	surprise	none
F-measure	0,86033	0,6589	0,6280	0,7207	0,5932	0,6462

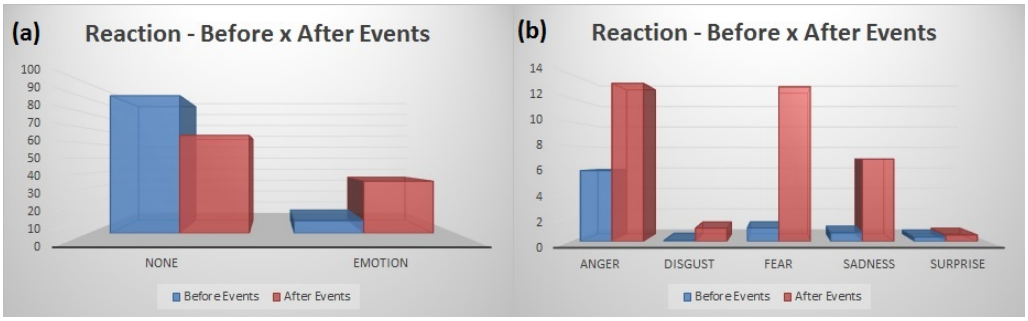


Figure 1. Tweets distribution before events and after events.

sidered our model reliable because it achieved average precision and recall above 70%, which we believe were good results taking into account results presented in (Suttles and Ide 2013; Purver and Battersby 2012; Mohammad et al. 2015). F-measure results for such a model can be seen in Table 5. It can be seen that the model's result for anger stands out along with the one for sadness. Remaining emotions have similar results, excluding surprise that performed below 60% but still close to the average.

5. Analysis

The first question we wanted to answer with our dataset was if there exists an emotion shift due to terrorism events. To answer this question, we compared emotion distribution before the events (BM and BL), and after them (AM and AL, respectively). Figure 1 depicts this comparison, where Y axis represents the percentage with regard to the total number of tweets of the respective dataset. All tweets are considered in Figure 1.(a), whereas only tweets with emotion are shown in Figure 1.(b). The first result observed was that before the events just about 8% of the tweets contained emotions from our emotion categories while after the events that number increased to around 25%. Furthermore, three emotions prevailed after the events: anger, fear, sadness. No significant changes were observed for disgust and surprise. Therefore, we conclude that there is indeed an emotional shift due to terrorism events.

The second question was whether different terrorism events raise the same emotional reaction. To answer this question we compared AM and AL in terms of emotion distribution. Figure 2 depicts emotion distribution for both events, where the Y axis represents, for each class (#prayformanchester and #londonbridge), the percentage of its total number of tweets distributed in emotion categories. Only tweets with emotion are shown. The results reveal that there are differences between these two events. While the event in London raised anger in the majority, the one in Manchester raised in the majority fear, followed by sadness.

A demographic analysis helped us understanding the differences between these two events. Figure 3 depicts emotion distribution by gender and age. Figure 4 shows gender and age distribution for both events. The Y axis represents, for each class (Gender

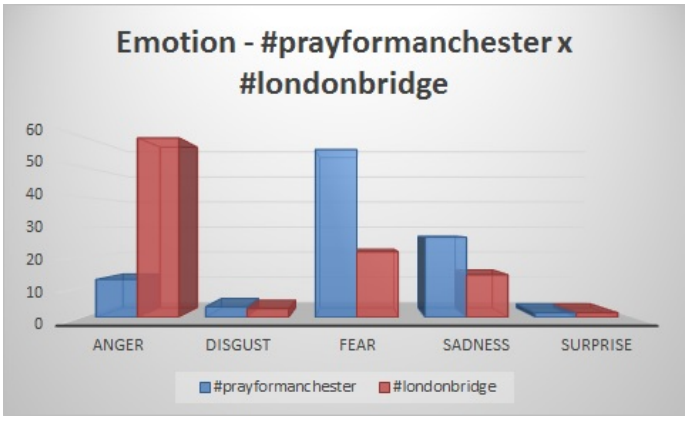


Figure 2. Emotion distribution for terrorism events.

Figure 3. Emotion distribution by Age and Gender.

and/or Age), the percentage of its total number of tweets per category for each dataset. In all graphs, tweets without emotion were not shown. The distribution of tweets by emotions for genders showed that Female users feel more fear and sadness compared to Male ones, who feel more anger instead. In addition, the distribution of tweets by emotion for ages shows that as the age increases, the feeling of anger increases proportionally. Fear, on the other hand, is higher for young ages, and it smoothly drops as age increases. No particular behavior was observed with regard to demographics for sadness. With these results, we distributed age and gender for both events and observed that there are indeed differences due to the concerned audience. In the Manchester event, the majority of tweeters are young women, i.e. the exact profile who mostly feels fear. This can be explained by the fact that Ariana Grande is very popular in this demographics. In the London event, such distribution showed that the majority of the tweeters were male middle-aged or older, i.e. the exact profile who feels anger. We believe that London

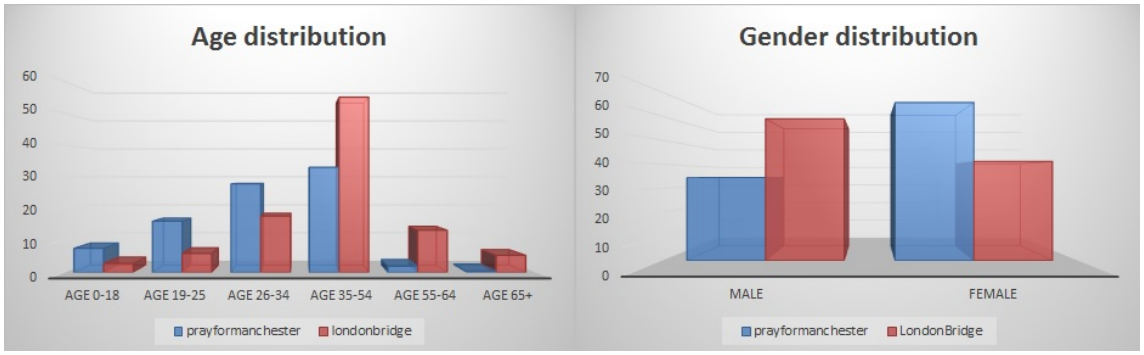


Figure 4. Tweet distribution by Gender and Age for both events.

Figure 5. Tweet distribution by Location for both events.

bridge have affected the average London citizen who could be potentially at the location of the attack. Thus, we conclude that each terrorism event may raise distinct predominant emotions. Our hypothesis, to be confirmed, is that it is related to the people who see themselves as potential victims of a similar attack.

The third question was whether location has an impact on emotional reaction. To answer this question, we compared tweets from UK and US. The distributions for each country are depicted in Figure 5, where Y axis represents, for each class (UK and US), the percentage of its total number of tweets distributed in emotion categories. Only tweets with emotions are considered. For both locations, the distribution of tweets into emotion categories for both events did not show any noticeable variation. These findings indicate that location may not be an important factor as much as age and gender are.

6. Conclusion

Our work provided a study on the emotional reaction of twitter users to terrorism events. We addressed negative emotions and used deep learning approach for emotion prediction. Demographic data such as location, age and gender were extracted with help of available tools. Our results showed that when terrorism events occur, a shift of emotion towards anger, sadness and fear can be noticed. In addition, our demographic analysis showed that gender and age have influence on how tweeters react to terrorism events. Our data indicated that young Women tend to feel fear and sadness while Man in middle age and above tend to feel anger. Location did not provide any noticeable impact on the emotional reaction.

As contribution, we derive an emotion dataset in the context of terrorism and provided a CNN model that achieved good performance for emotions in our context. The questions we answered were a first step towards understanding the emotional reaction terrorism events raise on general population. We hope our work encourage further studies on social media focusing on terrorism, which we believe impact people in a complex emotional way. The data we provided might be used for further analysis and the results we reported might be used to better developing specific assistance programs for coping with terror. One opportunity is to improve our work by selecting similar terrorism attacks, as the differences of our targeted events might bring some noise to our analysis when comparing them. Another opportunity is to consider the location as indicated by georeferenced tweets and then study its possible impact. This because even if georeferenced tweets constitute a small set of the total, their information may be more accurate then the ones filtering by the location indicated in the users' profile.

References

[Ain et al. 2017] Ain, Q. T., Ali, M., Riaz, A., Noureen, A., Kamran, M., Hayat, B., and Rehman, A. (2017). Sentiment analysis using deep learning techniques: A review. *International Journal of Advanced Computer Science and Applications*, 8(6).

[Anderson 2005] Anderson, B. (2005). Imagined communities. *Chap*, 4(Hansen 1999):48–60.

[Bravo-Marquez et al. 2016] Bravo-Marquez, F., Frank, E., Mohammad, S. M., and Pfahringer, B. (2016). Determining word-emotion associations from tweets by multi-label classification. In *Proc. of the IEEE/WIC/ACM WI*, pages 536–539.

[Burnap et al. 2014] Burnap, P., Williams, M. L., Sloan, L., Rana, O., Housley, W., Edwards, A., Knight, V., Procter, R., and Voss, A. (2014). Tweeting the terror: modelling the social media reaction to the Woolwich terrorist attack. *Social Network Analysis and Mining*, 4(1):1–14.

[De Choudhury et al. 2016] De Choudhury, M., Jhaver, S., Sugar, B., and Weber, I. (2016). Social media participation in an activist movement for racial equality. In *Proc. of the ICWSM*, pages 92–101.

[Ekman and Friesen 1982] Ekman, P. and Friesen, W. (1982). Emotion in the human face system. *Cambridge University Press, San Francisco, CA,*.

[ElSherief et al. 2017] ElSherief, M., Belding, E. M., and Nguyen, D. (2017). # notokay: Understanding gender-based violence in social media. In *Proc. of the ICWSM*, pages 52–61.

[Fan et al. 2014] Fan, H., Cao, Z., Jiang, Y., Yin, Q., and Doudou, C. (2014). Learning deep face representation. *CoRR*.

[Gallegos et al. 2015] Gallegos, L., Lerman, K., Huang, A., and Garcia, D. (2015). Geography of emotion: Where in a city are people happier? In *Proc. of the WWW*, pages 569–574.

[Go et al. 2009] Go, A., Bhayani, R., and Huang, L. (2009). Twitter Sentiment Classification using Distant Supervision. *Processing*, 150(12):1–6.

[Hasan et al. 2014] Hasan, M., Rundensteiner, E., and Agu, E. (2014). EMOTEX: Detecting Emotions in Twitter Messages. *ASE BIGDATA/SOCIALCOM/CYBERSECURITY Conference*, pages 27–31.

[Horgan 2014] Horgan, J. (2014). *The Psychology of Terrorism, Second Edition*. Taylor & Francis Group.

[Kim et al. 2011] Kim, S., Bak, J., Jo, Y., and Oh, A. (2011). Do You Feel What I Feel ? Social Aspects of Emotions in Twitter Conversations. *NIPS Workshop*, pages 495–498.

[Kim 2014] Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proc. of EMNLP*, pages 1746–1751.

[Lerman et al. 2016] Lerman, K., Arora, M., Gallegos, L., Kumaraguru, P., and Garcia, D. (2016). Emotions, demographics and sociability in twitter interactions. In *Proc. of the ICWSM*, pages 201–210.

[Liu 2012] Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5(1):1–167.

[Lotan et al. 2011] Lotan, G., Graeff, E., Ananny, M., Gaffney, D., Pearce, I., and danah boyd (2011). The arab spring— the revolutions were tweeted: Information flows during the 2011 tunisian and egyptian revolutions. *International Journal of Communication*, 5(0).

[Mitchell et al. 2013] Mitchell, L., Frank, M. R., Harris, K. D., Dodds, P. S., and Danforth, C. M. (2013). The geography of happiness: Connecting twitter sentiment and expression, demographics, and objective characteristics of place. *PLOS ONE*, 8(5):1–15.

[Mohammad 2012] Mohammad, S. (2012). #Emotional Tweets. In *Proc. of the First Conference on Lexical and Computational Semantics*, pages 246–255.

[Mohammad and Turney 2013] Mohammad, S. M. and Turney, P. D. (2013). Crowdsourcing a word-emotion association lexicon. 29(3):436–465.

[Mohammad et al. 2015] Mohammad, S. M., Zhu, X., Kiritchenko, S., and Martin, J. (2015). Sentiment, emotion, purpose, and style in electoral tweets. 51(4):480–499.

[Munezero et al. 2014] Munezero, M. D., Montero, C. S., Sutinen, E., and Pajunen, J. (2014). Are they different? affect, feeling, emotion, sentiment, and opinion detection in text. *IEEE Transactions on Affective Computing*, 5(2):101–111.

[Purver and Battersby 2012] Purver, M. and Battersby, S. (2012). Experimenting with Distant Supervision for Emotion Classification. *Proc. of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 482–491.

[Sakaki et al. 2010] Sakaki, T., Okazaki, M., and Matsuo, Y. (2010). Earthquake shakes twitter users: Real-time event detection by social sensors. In *Proc. of the 19th International Conference on World Wide Web*, pages 851–860.

[Suttles and Ide 2013] Suttles, J. and Ide, N. (2013). Distant supervision for emotion classification with discrete binary values. In Gelbukh, A., editor, *Computational Linguistics and Intelligent Text Processing*, pages 121–136, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Wan and Paris 2015] Wan, S. and Paris, C. (2015). Understanding Public Emotional Reactions on Twitter. *Proc. of ICWSM*, pages 715–716.

[Wang et al. 2012] Wang, W., Chen, L., Thirunarayan, K., and Sheth, A. P. (2012). Harnessing twitter "big data" for automatic emotion identification. In *2012 International Conference on Privacy, Security, Risk and Trust*, pages 587–592.

PrivLBS: Uma Abordagem para Preservação de Privacidade de Dados em Serviços baseados em Localização

Eduardo R. D. Neto[1], André L. C. Mendonça[1], Felipe T. Brito[1], Javam C. Machado[1]

[1]Laboratório de Sistemas e Banco de Dados (LSBD)
DC/UFC – UFC – CEP 60440-900 – Fortaleza – CE – Brazil

{eduardo.rodrigues,andre.luis,felipe.timbo,javam.machado}@lsbd.ufc.br

Abstract. *Location based services have been increasingly integrated into people's daily activities. However, some of these services may not be trustworthy and lead to serious privacy breaches. This work proposes a new technique for privacy preserving data, named PrivLBS, which ensures that individual's location will not be easily re-identified by malicious services. Experimental results show that, for euclidean distance-based attacks, individual's probability of location re-identification, after using PrivLBS, is around 11.4%, whereas in existing work, this probability reaches 59.2%.*

Resumo. *Serviços baseados em localização têm sido integrados às atividades diárias das pessoas. Entretanto, alguns desses serviços podem não ser confiáveis e levar a sérios riscos de violação de privacidade. Este trabalho propõe uma nova técnica de preservação de privacidade de dados, denominada PrivLBS, capaz de assegurar que as localizações dos indivíduos não serão facilmente reidentificadas por serviços mal intencionados. Resultados de avaliação experimental demonstram que, para ataques baseados em distância euclidiana, a probabilidade de reidentificação das localizações de um indivíduo, após utilização do PrivLBS, é em torno de 11.4%, enquanto que, em trabalhos já existentes na literatura, essa probabilidade chega a 59.2%.*

1. Introdução

Serviços baseados em localização (*Location-Based Services, LBS*) são serviços que proveem recursos adicionais a dispositivos móveis baseado em suas localizações geográficas. Esses serviços têm sido integrados às atividades diárias das pessoas, permitindo que elas utilizem sua localização atual para diversos fins, tais como navegação, rastreamento, recomendação, entre outros. Em geral, para que serviços baseados em localização sejam utilizados, os usuários enviam ao provedor de serviço (provedor de LBS) sua identidade e localização geográfica real, definida pela latitude e longitude, além de consultas que se desejam obter respostas, como o shopping mais próximo, supermercado, restaurante [Niu et al. 2014]. Dessa forma, os usuários obtém os locais relativos à consulta realizada.

Por outro lado, a utilização de serviços baseados em localização pode levar a sérios riscos de violação de privacidade devido a provedores de serviços mal intencionados ou não confiáveis [Li et al. 2014, Niu et al. 2015]. Provedores de LBS não confiáveis são capazes de expor dados de localização de seus usuários ou até mesmo vender informações de localizações a terceiros [Zhu et al. 2013]. De posse dessas informações, os dados obtidos por terceiros são utilizados para descoberta de padrões de movimento do usuário,

podendo revelar informações sensíveis sobre ele. Por exemplo, se um usuário, ao utilizar um serviço baseado em localização, geralmente exibe sua localização próximo a um hospital, as informações de localização poderiam ser utilizadas para inferir que aquele usuário provavelmente possa ter algum problema de saúde.

Para que seja mantida a privacidade dos usuários na utilização desses serviços, várias técnicas de preservação de privacidade em LBS foram propostas nos últimos anos [Niu et al. 2016, Tsoukaneri et al. 2016, Ullah and Shah 2016, Sun et al. 2017b]. Algumas dessas técnicas são baseadas em métodos de camuflagem, os quais empregam o modelo de privacidade k-anonimato [Sweeney 2002] para proteger a privacidade dos locais percorridos por um usuário. Este modelo garante que um usuário só poderá ser reidentificado com probabilidade $\frac{1}{k}$, onde k é o grau de privacidade especificado pelo usuário. Quanto maior o valor de k, menor a probabilidade de reidentificação das localizações de um indivíduo.

Uma forma de camuflar as localizações de um usuário, utilizando o modelo de privacidade k-anonimato, é por meio da técnica de *"dummy locations"* [Kido et al. 2005]. Nessa abordagem, $k-1$ localizações falsas são geradas e adicionadas à consulta realizada pelo usuário ao provedor do LBS, a fim de confundir a localização real do indivíduo que realizou a consulta. Por exemplo, no momento em que um usuário desejar obter o shopping mais próximo de sua localização atual, ao especificar o valor de k, outras $k-1$ localizações falsas serão geradas e enviadas ao provedor de serviço. O provedor retornará ao usuário os shoppings mais próximos para cada umas das $k-1$ localizações falsas, como também o shopping mais próximo da localização real do usuário. Contudo, trabalhos existentes na literatura [Kido et al. 2005, Vu et al. 2012, Niu et al. 2014, Sun et al. 2017a] não levam em consideração qualquer métrica de distância física das localizações no momento da geração, o que as tornam vulneráveis a ataques que exploram essa deficiência. Assim, localizações falsas geradas podem não ser coerentes com a distância percorrida pelo usuário durante o intervalo de realização de duas consultas consecutivas.

Assumindo que o provedor do LBS não é confiável, neste trabalho propomos uma nova técnica baseada no modelo de privacidade k-anonimato, denominada PrivLBS, capaz de assegurar que as localizações dos indivíduos que utilizam serviços baseado em localização não serão facilmente reidentificadas. Para isso, propomos um novo tipo de ataque baseado em distância que busca revelar a localização real do usuário, considerando a distância euclidiana entre as localizações de consultas consecutivas enviadas ao provedor de serviço. Demonstramos, através de simulações, que nosso modelo de ataque possui uma alta taxa de reidentificação das localizações reais dos usuários quando aplicado sobre a estratégia DLP (*Dummy Location Privacy-preserving*) [Sun et al. 2017a], proposta recentemente na literatura. Por outro lado, PrivLBS assegura que provedores de serviços não confiáveis, que utilizam ataques baseado em distância, não são capazes de violar a privacidade dos usuários com probabilidade média maior que $\frac{1}{k}$, onde k é o grau de privacidade.

O restante do artigo está organizado da seguinte forma. A Seção 2 apresenta os trabalhos relacionados ao tema de preservação de privacidade em serviços baseados em localização. Na Seção 3 apresentamos o nosso modelo de ataque baseado em distância euclidiana. Em sequência, na Seção 4, apresentamos o método PrivLBS como solução para o problema e, em seguida, o avaliamos experimentalmente na Seção 5 utilizando um

conjunto de dados real. Por fim, a Seção 6 conclui o trabalho e apresenta os direcionamentos futuros de pesquisa.

2. Trabalhos Relacionados

Diversas soluções foram propostas com o objetivo de garantir a privacidade de usuários ao utilizarem serviços baseados em localização e, assim, impedir que suas informações sensíveis sejam descobertas. Em sua grande maioria, as soluções são divididas em abordagens baseadas em anonimização de localizações [Gedik and Liu 2008, Ying and Makrakis 2014], criptografia [Lu et al. 2014] ou seleção de *dummy locations* [Niu et al. 2014, Niu et al. 2015, Sun et al. 2017a].

O trabalho em [Gedik and Liu 2008] propõe um modelo personalizado do *k*-anonimato utilizando a estratégia de camuflagem. Nesse trabalho, os autores utilizaram um servidor de anonimização confiável que considera o *trade-off* entre a privacidade da localização e a qualidade do serviço para anonimizar a localização dos usuários. Na solução, uma região de camuflagem contendo outros $k-1$ usuários, geograficamente distribuídos, é formada e, somente então, a consulta é submetida ao serviço baseado em localização. Também utilizando a estratégia de camuflagem, o trabalho em [Ying and Makrakis 2014] assegura a privacidade dos usuários ao construir uma região de camuflagem contendo, pelo menos, *k* usuários e *l* segmentos de rua.

O trabalho proposto em [Lu et al. 2014] apresenta um *framework*, denominado PLAM, para a preservação de privacidade em redes sociais de área local. Esse *framework*, além de atender ao modelo de privacidade *k*-anonimato, também assegura o modelo *l*-diversidade [Machanavajjhala et al. 2006], considerando casos em que um adversário pode inferir informações sensíveis sobre indivíduos mesmo sem identificá-los. Entretanto, o servidor de anonimização confiável é substituído por uma técnica de criptografia, denominada pseudo-ID, a qual não mantém a utilidade dos dados para fins de análise.

[Niu et al. 2014] propõem o DLS (*Dummy Location Selection*), um algoritmo de seleção de *dummy locations* baseado em entropia, o qual mede o grau de incerteza sobre um conjunto de localizações selecionadas. Nesse trabalho, os autores apresentaram um modelo de LBS no qual o provedor do serviço é responsável por coletar e disponibilizar aos usuários dados estatísticos sobre as consultas. Tais dados dizem respeito às probabilidades nas quais requisições são demandadas ao LBS. Assim, o DLS assegura a privacidade dos usuários, garantindo as propriedades do modelo *k*-anonimato, ao submeter uma consulta contendo a localização real do usuário e de outras $k-1$ localizações falsas escolhidas utilizando como critério de seleção localizações que tenham uma probabilidade de ser enviada ao LBS semelhante a da localização real.

Por fim, o trabalho em [Sun et al. 2017a] propõe o algoritmo DLP, que assim como o DLS utiliza a técnica de *dummy locations* e a probabilidade das localizações sobre as consultas feita ao LBS como critério de seleção, porém alcançando um grau de entropia superior aos trabalhos anteriores, isto é, uma maior incerteza sobre um conjunto de localizações selecionadas. Os autores propõem um algoritmo de ataque desenvolvido especificamente para revelar a localização real do usuário quando a anonimização utiliza como critério de seleção das $k-1$ localizações falsas a probabilidade destas nas consultas enviadas e coletadas pelo LBS.

Ao contrário das soluções anteriores, este artigo propõe uma técnica baseada na

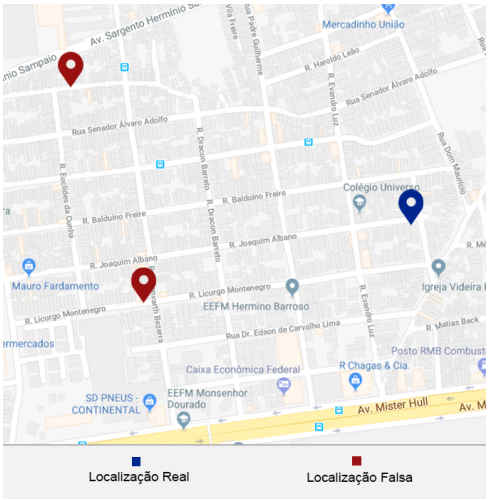
seleção de *dummy locations* cujas localizações falsas são selecionadas utilizando critérios de distância euclidiana e probabilidade de suas ocorrências com base em informações coletadas pelo LBS, garantindo, assim, uma maior privacidade aos usuários contra ataques que explorem esses critérios sobre as consultas enviadas ao LBS.

3. Ataque baseado em Distância Euclidiana

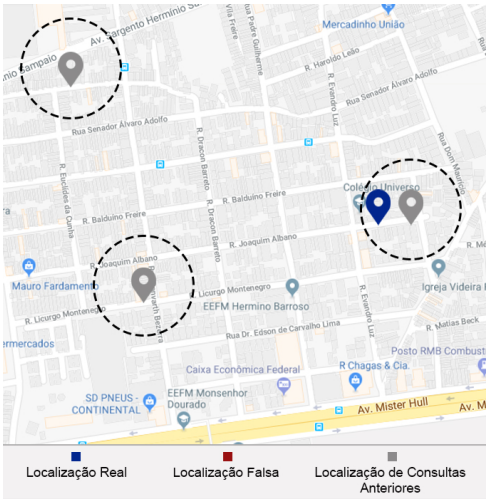
Quando lidamos com serviços baseados em localização, pode-se realizar dois tipos de requisições (consultas) ao provedor de serviço: consultas simples e contínuas. Uma consulta simples consiste em uma requisição realizada pelo usuário antes mesmo dele obter um novo identificador, por exemplo, quando um usuário solicita o shopping mais próximo da localização informada e, após receber o conteúdo requisitado, encerra a conexão com o LBS. Caso seja realizada uma nova requisição ao LBS, o cliente já é visto como um novo usuário. Consultas contínuas tratam-se de múltiplas consultas realizadas por um usuário em um determinado intervalo de tempo por meio de um mesmo identificador. Por exemplo, quando um usuário solicita o tempo estimado para se chegar a um destino, várias vezes em um determinado intervalo de tempo, até que o mesmo encerre a requisição. Para qualquer tipo de consulta, o LBS recebe a requisição e retorna a informação requerida de acordo com seu conteúdo. Este trabalho visa preservar a privacidade de indivíduos que realizam tanto consultas simples quanto consultas contínuas a provedores de LBS.

Para demonstrar a eficiência do PrivLBS em relação aos modelos existentes de geração de localizações falsas, propomos um algoritmo de ataque baseado em distância que visa revelar a localização real do usuário utilizando a métrica de distância euclidiana entre as localizações de duas consultas consecutivas enviadas pelo mesmo usuário. Vale ressaltar que o algoritmo de ataque proposto pode ser adaptado para utilizar qualquer tipo de função de distância, não apenas a euclidiana.

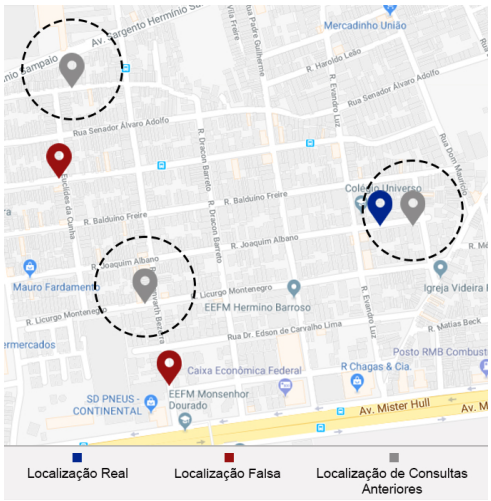
Utilizaremos a Figura 1 para ilustrar como o ataque é realizado sobre uma nova requisição feita ao LBS quando a anonimização não leva em consideração a distância euclidiana entre os pontos no momento da escolha de suas localizações falsas. Os pontos em azul e vermelho representam as localizações reais e falsas, respectivamente. Os pontos em cinza são as localizações da última consulta enviada ao LBS. Na Figura 1(a) observamos a requisição no primeiro momento t_0 anonimizada com grau de privacidade $k = 3$, escolhido especificamente para simplificar o exemplo. Já a Figura 1(b) apresenta o momento em que o usuário, após um intervalo de tempo t, realiza a consulta seguinte em uma nova localização. A circunferência ao redor dos pontos em cinza representam a área contendo todos os pontos alcançáveis a partir dele. A Figura 1(c) exibe as localizações selecionadas no processo de anonimização e enviadas na requisição ao LBS pelo usuário no tempo t_1. O algoritmo de ataque, tendo obtido o domínio da consulta anterior enviada ao LBS, verifica quais pontos da nova consulta estão dentro de uma das áreas de alcance dos pontos da consulta anterior. As localizações da nova consulta que estiverem dentro dessas áreas são as localizações candidatas, visto que as outras localizações devem ser ignoradas por não serem alcançáveis pelo usuário no intervalo de tempo de realização de consultas consecutivas. O algoritmo de ataque seleciona como localização real uma das localizações dentre as candidatas. Podemos observar pela Figura 1(c) que apenas um ponto da nova consulta está dentro de uma dessas áreas, sendo assim identificada pelo algoritmo de ataque como a localização real do usuário.



(a) Requisição em t_0.



(b) Nova localização em t_1.



(c) Requisição anonimizada em t_1.

Figura 1. Anonimização de localizações sem critério de distância.

O Algoritmo 1 refere-se à nossa proposta de ataque, que possui como entrada os parâmetros R', R, denotando respectivamente o conjunto das localizações contidas na requisição anterior e atual. O parâmetro P, contendo a lista de localizações atendidas pelo LBS e suas respectivas probabilidades. Além disso, o algoritmo tem como parâmetro de entrada um $limite$, estabelecido pelo atacante, que representa a distância máxima permitida entre as localizações. Esse parâmetro é calculado pela função $limite = v * t$, onde v é a velocidade média do usuário estimada pelo LBS, e t é o tempo decorrido entre uma requisição e outra.

O algoritmo atua da seguinte forma: para cada localização r_i da nova requisição R, o algoritmo calcula a distância euclidiana entre r_i e cada uma das localizações r'_j da requisição anterior R'. Se essa distância for menor ou igual ao $limite$, então a localização r_i é adicionado ao conjunto das localizações candidatas C. Quanto mais preciso o $limite$, mais eficaz é o algoritmo, visto que ele define os elementos do conjunto das localizações candidatas C à localização real. Um $limite$ alto implica em um relaxamento da condição

de alcançabilidade de um ponto a outro, aumentando a probabilidade de localizações que não são realmente alcançáveis serem adicionadas ao conjunto C e, portanto, diminuindo a precisão do algoritmo. De maneira análoga, um limite baixo implica em uma restrição maior na escolha dos elementos do conjunto C, o que leva a um conjunto com poucos elementos. Por fim, o algoritmo de ataque proposto retorna, como localização real, a localização l com maior probabilidade, conforme P, dentre aquelas do conjunto C.

Algoritmo 1: ATAQUE BASEADO EM DISTÂNCIA EUCLIDIANA

```
Entrada: R' , R, limite, P
Saída: l
para cada localização r_i ∈ R faça
    para cada localização r'_j ∈ R' faça
        se Distância(r_i, r'_j) <= limite então
            Insere r_i em C;
        fim
    fim
fim
l = maxProb(r ∈ C);
retorna l
```

4. PrivLBS

Para contornar o problema da preservação de privacidade de dados de um usuário, que utiliza um serviço baseado em localização, adotamos um modelo semelhante ao proposto em [Sun et al. 2017a]. Primeiramente, o LBS é responsável por coletar, para cada localização l_i, a probabilidade q_i de uma consulta sobre ela. Tal probabilidade é definida pela Equação 1, denominada informação complementar (*side information*).

$$q_i = \frac{\textit{número de consultas sobre } l_i}{\textit{número total de consultas}} \tag{1}$$

A Figura 2 ilustra o fluxo da abordagem proposta. Inicialmente, o usuário inicia a sessão requisitando ao LBS a informação complementar coletada. Após obter essa informação, a anonimização da consulta é realizada utilizando o Algoritmo 2, que seleciona $k-1$ localizações falsas a serem adicionadas à consulta. Dessa forma, o provedor do LBS irá responder conforme o conteúdo da requisição. Por fim, o usuário filtra aquela informação que é de seu interesse.

Detalhando o processo de anonimização. PrivLBS recebe como parâmetros de entrada o grau de privacidade k, a informação complementar P, contendo a lista de localizações atendidas pelo LBS e suas respectivas probabilidades, a localização real l_r e a última requisição enviada R'. O usuário armazena, em seu histórico, a última requisição enviada ao LBS. Caso o histórico do usuário esteja limpo, isto é, o usuário está fazendo a sua primeira ou única consulta, o parâmetro R' será nulo. Neste caso, a seleção das localizações falsas é feita utilizando o próprio algoritmo DLP, proposto por [Sun et al. 2017a]. Nesta situação o algoritmo de ataque baseado em distância não é aplicável, já que não há uma consulta anterior. Caso R' não seja nulo, para cada

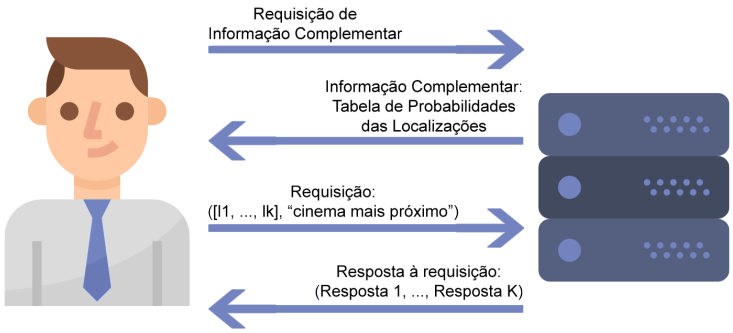


Figura 2. Fluxo de informações do PrivLBS.

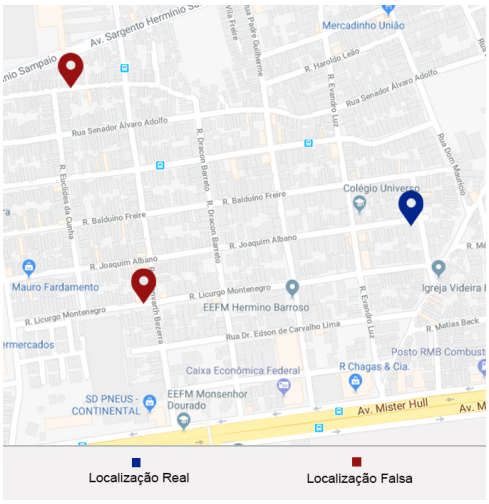
localização falsa $r'_i \in R'$, é construído um conjunto A_i, contendo todas as localizações alcançáveis a partir de r'_i, obtidas através da função *BuscarDistância(P,r'_i)*. Essa função calcula a distância euclidiana entre as localizações e, caso a distância seja menor que a distância máxima possível de ser percorrida pelo usuário, ela é adicionada ao conjunto. De cada conjunto A_i é selecionada a localização cuja probabilidade mais se aproxima da localização real l_r, adicionando-as ao conjunto L, formando $k-1$ localizações falsas. Tais localizações são adicionadas à localização real l_r em L e enviadas ao servidor do LBS.

Algoritmo 2: PRIVLBS

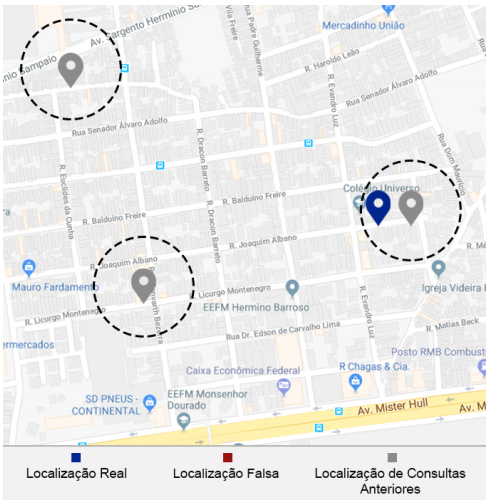
Entrada: k, P, l_r, R'
Saída: L
1 **se** *R' == vazio* **então**
2 $L \leftarrow DLP(k, l_r)$;
3 **senão**
4 **para cada** $r'_i \in R'$ **faça**
5 $A_i \leftarrow BuscarDistância(P, r'_i)$;
6 *Insere em L a localização contida em* A_i *cuja probabilidade seja a mais próxima de* l_r;
7 **fim**
8 *Insere* l_r *em L*
9 **fim**
10 **retorna** L

A Figura 3 ilustra como funciona o algoritmo PrivLBS. Novamente, os pontos em azul e vermelho representam as localizações reais e falsas, respectivamente. Os pontos em cinza são as localizações das consultas anteriores. A Figura 3(a) representa o momento inicial, onde o usuário realiza a primeira consulta anonimizada com grau de privacidade $k = 3$. A Figura 3(b) apresenta o momento seguinte, onde o usuário se desloca para uma nova posição após um intervalo de tempo t e realiza uma nova consulta. As circunferências ao redor dos pontos em cinza representam as áreas contendo todos os pontos alcançáveis a partir dos vértices e possíveis candidatos a serem selecionados pelo PrivLBS como localizações falsas. A Figura 3(c) mostra os pontos selecionados pelo algoritmo PrivLBS que irão fazer parte da requisição junto à localização real a ser enviada ao LBS. Como o PrivLBS seleciona, para cada localização da consulta anterior, uma localização que seja alcançável por ela, dado a velocidade do usuário e o intervalo

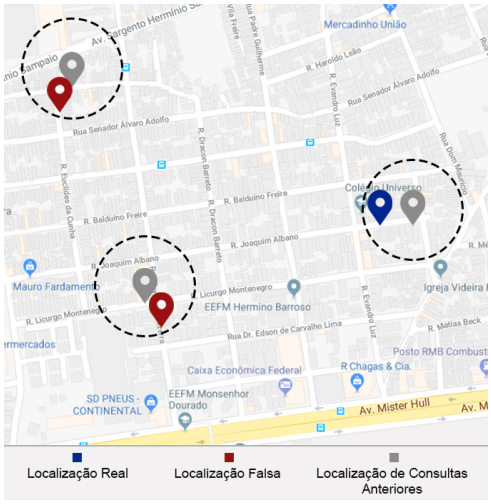
de tempo decorrido entre as consultas, um possível ataque que visa explorar esse critério observa cada uma das localizações na nova consulta como deslocamentos possíveis do usuário. Dessa forma, o atacante não é capaz de reidentificar a localização real com probabilidade superior a $\frac{1}{k}$. Isso garante o modelo de privacidade k-anonimato. Além disso, o algoritmo procura escolher localizações alcançáveis que tenham um probabilidade de consulta ao LBS semelhante à localização real, o que protege também o usuário contra ataques probabilísticos sobre o teor da consulta, isto é, ataques que visam identificar, na consulta, uma localização que tenha uma probabilidade maior que as outras.



(a) Requisição em t_0



(b) Nova localização em t_1



(c) Requisição anonimizada em t_1

Figura 3. Anonimização de localizações utilizando o algoritmo PrivLBS.

5. Experimentos

Foram realizados experimentos a fim de avaliar a eficácia do algoritmo PrivLBS frente a ataques baseados em distância. Nossa análise foi realizada com base na taxa de reconhecimento da localização real quando aplica-se o ataque baseado em distância sobre a consulta. Nós também mensuramos o grau de privacidade da requisição, denotado por sua entropia, que consiste na incerteza de identificação da localização real dentre as localizações

falsas selecionadas [Serjantov and Danezis 2003], independente da distância. Quanto maior a entropia, mais incerta é a informação acerca das localizações.

5.1. Conjunto de dados

Utilizamos um conjunto de dados real disponibilizado pela CTA[1] (*Chicago Transit Authority*), responsável por operar o segundo maior sistema de transporte público dos Estados Unidos, atendendo toda a cidade de Chicago e 35 subúrbios na periferia dessa cidade. Esse conjunto de dados foi escolhido por conter, para cada uma das 11.593 estações de ônibus, além da latitude e longitude, a média de embarque em um dia de semana do mês de Outubro de 2012. Isso nos permitiu estimar a probabilidade de requisições sobre cada uma das estações de ônibus com base na média de embarque, formando assim a informação complementar sobre as localizações utilizada tanto nos algoritmos de anonimização DLP (nosso *baseline*) e PrivLBS, como também no algoritmo de ataque baseado em distância e no algoritmo de ataque baseado em probabilidade, proposto em [Sun et al. 2017a].

5.2. Simulação

Foram simulados dez mil usuários realizando consultas consecutivas ao LBS. Para cada usuário foi selecionada uma posição inicial aleatória do conjunto de dados. Cada usuário realizou três consultas consecutivas utilizando os algoritmos de anonimização PrivLBS e DLP. Cada consulta foi realizada em um momento temporal (e.g. t_0, t_1 e t_2). Entre os momentos t_0 e t_1, t_1 e t_2, foi simulado um deslocamento para alguma localização aleatória que se encontra dentro de um raio de 1 km da localização anterior. Estabeleceu-se esse limite considerando um intervalo de 1 minuto entre uma consulta e outra, e uma velocidade média de 60 km/h do usuário, resultando em um deslocamento de até 1 km. Ao término de cada consulta calculamos a entropia sobre o conjunto de localizações selecionadas e aplicamos o algoritmo de ataque baseado em distância e o algoritmo de ataque baseado na probabilidade de execução das consultas sobre cada localização para simular um ataque do provedor de serviço ao tentar violar a privacidade de localização do usuário.

5.3. Resultados

Para demonstrar o grau de privacidade alcançado pelo PrivLBS, uma série de mil simulações foram realizadas, onde foram medidas a entropia sobre os conjuntos de localizações selecionadas e a probabilidade de reindentificação da localização real do usuário sobre vários graus de privacidade (valores de k).

Entropia	Grau de anonimização k				
	2	**4**	**8**	**16**	**32**
DLP	0,69	1,38	2,07	2,77	3,46
PrivLBS	0,58	1,26	1,92	2,60	3,28

Tabela 1. Comparação da entropia entre os algoritmos DLP e PrivLBS.

A Tabela 1 mostra a entropia média obtida utilizando os algoritmos DLP e PrivLBS na seleção das localizações falsas da consulta, variando o grau de privacidade k. Observa-se um comportamento constante, no qual o DLP apresentou uma entropia um

[1]http://www.transitchicago.com

pouco maior em todos os graus analisados. Isso implica que o DLP é menos suscetível a ataques que exploram a probabilidade das localizações contidas nas consultas realizadas ao provedor do LBS. Este comportamento já era esperado, visto que o PrivLBS constrói um subconjunto, baseado na distância, das localizações disponíveis, diminuindo a probabilidade de selecionar localizações com probabilidade semelhante à localização real.

Apesar disso, quando vamos calcular a taxa de reconhecimento da localização real, obtida utilizando o algoritmo de ataque proposto por [Sun et al. 2017a], percebe-se, conforme Figura 4, que tanto o algoritmo DLP como o algoritmo PrivLBS são robustos para este tipo de ataque, garantindo a propriedade do modelo *k*-anonimato, uma vez que, para qualquer grau de privacidade k no gráfico, a taxa de reconhecimento ficou abaixo de $\frac{1}{k}$.

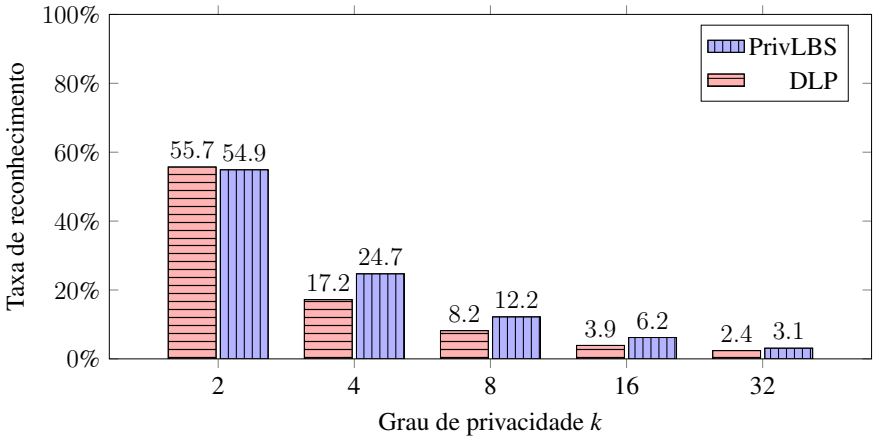


Figura 4. Taxas de reconhecimento da localização real para o ataque baseado na probabilidade de execução das consultas.

Já em relação a taxa de reconhecimento da localização real quando a consulta realizada sobre o provedor do LBS recebe o ataque baseado em distância euclidiana, podemos observar, conforme Figura 5, que quanto maior o grau de privacidade k, menor é a taxa de reconhecimento nas consultas realizadas utilizando o DLP como algoritmo de anonimização. Vale ressaltar que o parâmetro k representa, também, a quantidade de localizações que serão enviadas na consulta, aumentando a chance de mais localizações alcançáveis serem selecionadas como localizações falsas pelo algoritmo DLP. Este comportamento pode estar relacionado diretamente com o tamanho do conjunto de dados, já que isto aumentaria as chances de serem escolhidas localizações não alcançáveis pelo DLP. Apesar disso, podemos perceber que, para qualquer grau de privacidade k, o DLP não garante um *k*-anonimato, uma vez que a taxa de reconhecimento se manteve acima de $\frac{1}{k}$ para o ataque baseado em distância euclidiana.

Em contrapartida, nas requisições que utilizam o PrivLBS como algoritmo de anonimização, a taxa de reconhecimento se manteve sempre abaixo de $\frac{1}{k}$ para todos os graus de privacidade observados. Além disso, quando comparado ao algoritmo DLP, a probabilidade de reidentificacação das localizações de um usuário utilizando o PrivLBS é, em média, quatro vezes menor que o algoritmo DLP para os valores de $k = \{2, 4, 8, 16, 32\}$. Essa probabilidade chega a ser até cinco vezes menor que o algoritmo DLP quando $k = 8$, diminuindo o valor, que antes era de 59,2%, para 11,4%.

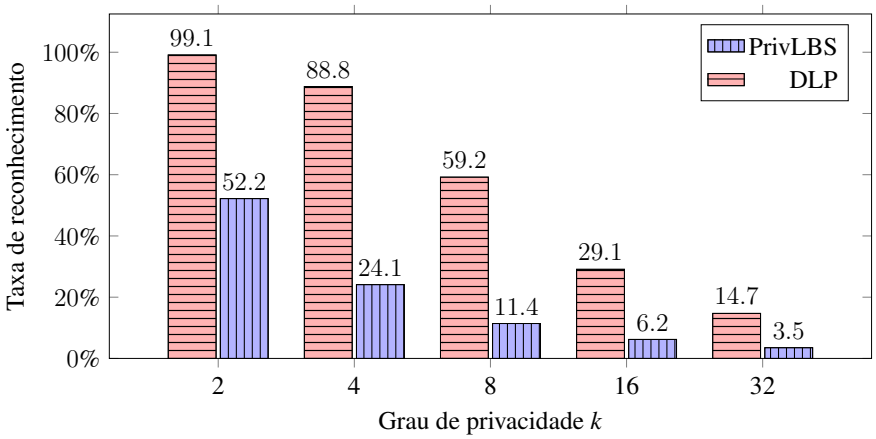


Figura 5. Taxas de reconhecimento da localização real para o ataque baseado em distância euclidiana.

6. Conclusão e Trabalhos Futuros

Neste trabalho apresentamos o PrivLBS, uma abordagem para preservação de privacidade de dados em serviços baseados em localização. Inicialmente propomos um modelo de ataque baseado na distância euclidiana entre as localizações contidas em requisições consecutivas ao LBS. Mostramos que esse tipo de ataque apresenta uma alta taxa de reidentificação quando aplicado sobre requisições consecutivas, que não foram anonimizadas considerando a distância euclidiana entre as localizações selecionadas. Demonstramos também que o PrivLBS, por ponderar tanto critérios de distância euclidiana como de probabilidade entre as localizações da consulta, apresenta uma baixa taxa de reidentificação ao sofrer ataques baseado em distância ou probabilísticos, garantindo as propriedades do modelo de privacidade k-anonimato.

Como trabalho futuro pretendemos realizar uma análise do impacto do tamanho do conjunto de dados sobre o PrivLBS, além de propor um modelo completo e dinâmico, buscando uma solução alternativa de seleção das localizações alcançáveis, que garanta uma maior entropia e produza o menor *overhead* possível, adotando, por exemplo, distância de rede de ruas para definir as localizações alcançáveis.

Agradecimentos

Os autores agradecem à CAPES, ao CNPq (132614/2017-0) e ao LSBD/UFC pelo financiamento parcial deste trabalho.

Referências

Gedik, B. and Liu, L. (2008). Protecting location privacy with personalized k-anonymity: Architecture and algorithms. *IEEE Transactions on Mobile Computing*, 7(1):1–18.

Kido, H., Yanagisawa, Y., and Satoh, T. (2005). An anonymous communication technique using dummies for location-based services. In *ICPS '05. Proceedings. International Conference on Pervasive Services, 2005.*, pages 88–97.

Li, H., Sun, L., Zhu, H., Lu, X., and Cheng, X. (2014). Achieving privacy preservation in wifi fingerprint-based localization. In *INFOCOM, 2014 Proceedings IEEE*, pages 2337–2345. IEEE.

Lu, R., Lin, X., Shi, Z., and Shao, J. (2014). Plam: A privacy-preserving framework for local-area mobile social networks. In *INFOCOM, 2014 Proceedings IEEE*, pages 763–771. IEEE.

Machanavajjhala, A., Gehrke, J., Kifer, D., and Venkitasubramaniam, M. (2006). l-diversity: Privacy beyond k-anonymity. pages 24–24.

Niu, B., Gao, S., Li, F., Li, H., and Lu, Z. (2016). Protection of location privacy in continuous lbss against adversaries with background information. In *2016 International Conference on Computing, Networking and Communications (ICNC)*, pages 1–6.

Niu, B., Li, Q., Zhu, X., Cao, G., and Li, H. (2014). Achieving k-anonymity in privacy-aware location-based services. In *INFOCOM, 2014 Proceedings IEEE*, pages 754–762. IEEE.

Niu, B., Li, Q., Zhu, X., Cao, G., and Li, H. (2015). Enhancing privacy through caching in location-based services. In *Computer Communications (INFOCOM), 2015 IEEE Conference on*, pages 1017–1025. IEEE.

Serjantov, A. and Danezis, G. (2003). Towards an information theoretic metric for anonymity. In Dingledine, R. and Syverson, P., editors, *Privacy Enhancing Technologies*, pages 41–53, Berlin, Heidelberg. Springer Berlin Heidelberg.

Sun, G., Chang, V., Ramachandran, M., Sun, Z., Li, G., Yu, H., and Liao, D. (2017a). Efficient location privacy algorithm for internet of things (iot) services and applications. *Journal of Network and Computer Applications*, 89:3 – 13. Emerging Services for Internet of Things (IoT).

Sun, G., Liao, D., Li, H., Yu, H., and Chang, V. (2017b). L2p2: A location-label based approach for privacy preserving in lbs. *Future Generation Computer Systems*, 74:375 – 384.

Sweeney, L. (2002). k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570.

Tsoukaneri, G., Theodorakopoulos, G., Leather, H., and Marina, M. K. (2016). On the inference of user paths from anonymized mobility data. In *2016 IEEE European Symposium on Security and Privacy (EuroS P)*, pages 199–213.

Ullah, I. and Shah, M. A. (2016). A novel model for preserving location privacy in internet of things. In *2016 22nd International Conference on Automation and Computing (ICAC)*, pages 542–547.

Vu, K., Zheng, R., and Gao, J. (2012). Efficient algorithms for k-anonymous location privacy in participatory sensing. In *2012 Proceedings IEEE INFOCOM*, pages 2399–2407.

Ying, B. and Makrakis, D. (2014). Protecting location privacy with clustering anonymization in vehicular networks. In *Computer Communications Workshops (INFOCOM WKSHPS), 2014 IEEE Conference on*, pages 305–310. IEEE.

Zhu, X., Chi, H., Niu, B., Zhang, W., Li, Z., and Li, H. (2013). Mobicache: When k-anonymity meets cache. In *Global Communications Conference (GLOBECOM), 2013 IEEE*, pages 820–825. IEEE.

Correlating educational documents from different sources through graphs and taxonomies

Márcio de Carvalho Saraiva[1], **Claudia Bauzer Medeiros**[1]

[1]Institute of Computing – University of Campinas
Caixa Postal 13083-852 – Campinas – SP – Brazil

{marcio.saraiva,cmbm}@ic.unicamp.br

Abstract. *Digital educational documents are growing in size and variety, and scientists are facing difficulties to find their way through them. One of the initiatives that have emerged to solve this problem involves the use of automatic classification algorithms. However, it is difficult to analyze implicit relationships among topics of materials. This paper presents CIMAL, a framework for enabling flexible access to material stored in arbitrary repositories. CIMAL combines semantic classification, taxonomies and graphs to elicit relationships among topics of educational documents. We validated our work using materials from Coursera (courses offered by Johns Hopkins University and University of Michigan) and a Higher Education Institute, from Brazil.*

1. Introduction

Usually, lecturers use educational material repositories to publish, store and share materials with their peers in academia and students. The access to those documents is usually open. Given such availability, how to find and choose the material(s) more suitable to study a given topic?

Sites such as the International Bank of Educational Objects [1], the ACM Learning Center and the ACM Techpack [2], the Coursera platform [3], MERLOT [4] and SlideShare [5] show that the access to collections of educational materials in different formats and the analysis of their contents are still done in a restricted way. Even simple queries through the interfaces of these repositories can result in a large number of items, making it difficult to understand them and select the relevant ones. Furthermore, none of these repositories offers means to analyze relationships among the stored objects, which would help select material. On the other hand, Web search engines return a set of potentially interesting documents, which may not be adapted to learning [Changuel et al. 2015].

Indeed, there has been a lack of solutions to identify topics in these materials and how they relate to others. Nevertheless, some efforts have emerged to help solving this problem, such as [Blei 2012, Rossi et al. 2015, Zhuang 2017] that try to discover, extract and collate large collections of thematic structures of documents. However, these and other solutions found in the literature have been conceived to classify documents based on training sets and annotations, strongly coupling the methods to a set of examples.

[1]http://objetoseducacionais2.mec.gov.br/
[2]http://learning.acm.org/, http://techpack.acm.org/cloud/
[3]https://www.coursera.org/
[4]http://www.merlot.org/
[5]http://www.slideshare.net/

Moreover, these solutions require extra tasks in addition to collecting the documents. Last but not least, such solutions have not been applied to sets with different formats of material and do not take advantage of other information from these materials to aid in the classification of topics.

Our proposal is a step towards helping people choose materials of interest from educational repositories. The problem handled in this paper is the elicitation and analysis of relations among different digital educational materials. Unlike related work, which concentrates only on textual sources, our methods process both slides and videos, extracts relevant topic and correlates them. In solving this problem we present the following contributions: (1) to reduce the effort to elicit relationships among various materials; (2) to specify and implement algorithms for correlation of educational material data (videos and slides) from different lecturers; (3) to enable users to conduct search on videos and slides to guide their studies.

This paper presents the design and implementation of CIMAL (Courseware Integration under Multiple relations to Assist Learning), abstractly presented in [Saraiva and Medeiros 2016]. CIMAL is a framework to analyze educational documents repositories, allowing visualizations of relationships among materials' topics through the use of graph algorithms. This work was validated with data from Johns Hopkins University and University of Michigan provided at Coursera, which is one of the largest e-learning repositories at the moment, and a Higher Education Institute from São Paulo - Brazil. Our work expands the analysis options in educational material repositories. Moreover, our proposal improves the search among different material formats by standardizing topics they cover.

2. Theoretical Foundation and Related Work

2.1. Educational Data Mining

According to Romero [Romero and Ventura 2013] Educational data mining is concerned with "researching, developing, and applying computerized methods to detect patterns in collections of educational data that would otherwise be hard or impossible to analyze due to the enormous volume of data within which they exist".

Typically, research towards helping users to select educational material can be roughly classified as (i) development of tools to analyze, access or store materials in repositories, (ii) mechanisms to integrate heterogeneous materials via user monitoring, and (iii) use of learning objects to encapsulate and standardize contents.

An example of (i), Ricarte et al. [Ricarte and Junior 2011] present a methodology to process data collected from educational environments to provide feedback to lecturers about the usage of the content they offer and to students about their behavior inside the environment. However, their work only provides information about access to a particular set of materials, and nothing is said about the content of these resources, the relationships between disciplines, teaching materials and topics mentioned.

An example of (ii) is the work of Little et al. [Little et al. 2012]. The authors look at the integration of multimedia search in the SocialLearn platform to assist users to build their own learning pathways by exploring and remixing content. The work emphasizes how content-based multimedia search technologies can be used to help lecturers and stu-

dents to find new materials and learning pathways by identifying semantic relationships between educational resources in a social learning network.

Finally, we can say that the set of slides and videos used in our research make up groups of learning objects, an example of (iii). According to Sathiyamurthy et. al [Sathiyamurthy et al. 2012] and the Institute of Electrical and Electronics Engineers (IEEE)[Learning Technology Standards Committee of the IEEE 2002] the notion of learning objects (LO) is recurrent in the context of research in EDM.

2.2. Components and Content from Educational Material

The strategy we adopted to extract and represent topics of educational material is inspired by a concept that we name *components of educational material*. Components are positional structures that highlight information of a given material in order to facilitate its understanding. Header, body, footer and numbering of slides are examples of components of slides; titles, subtitles and the progress bar are examples of components of videos. This information also can be used for analysis; in our work, we use these characteristics in classification, indexing, comparison and retrieval tasks.

Unlike other approaches in the literature that use the entire text of a document equally, we also extract information of components from different types of material to guide classification tasks. Our work presents a novel strategy for documents analysis, which considers the components present in the documents to facilitate the identification of topics in the documents.

2.3. Classification of topics

To classify educational materials, we use a technique called Explicit Semantic Analysis. In natural language processing and information retrieval, According to Egozi et al. [Egozi et al. 2011], Explicit Semantic Analysis (ESA) is semantic representation of text (entire documents or individual words) that uses a document corpus as a knowledge base. As described by [Gabrilovich and Markovitch 2009], ESA uses an association-based method that interprets a text segment by the strength of its association with concepts that are described in domain documents.

ESA assumes the availability of a vector of basic concepts, [C1, . . . , Cn], and represents each text fragment t by a vector of weights, [w1, . . . , wn], where wi represents the strength of association between t and Ci. Thus, the set of basic concepts can be viewed as a canonical n-dimensional semantic space, and the semantics of each text segment corresponds to a point in this space. This weighted vector is the semantic interpretation vector of t.

Such a canonical representation is very powerful, as it effectively allows us to estimate semantic relatedness of text fragments by their distance in this space.

2.4. Recognition of relationships

According to Jiang et al. [Jiang 2012], extraction of relations is the task of detecting and characterizing the semantic relations between entities in texts. They affirm that current state-of-the-art methods use carefully designed features or kernels and standard classification to solve this problem.

Mining of metadata (e.g., number of accesses to data or identification of entities in the documentation of objects) is often used to derive relationships among data, such as the work of Pereira[Pereira 2014]. Relationships of educational materials are viewed as the connections or associations among materials considering educational aspects, such as the association on the contents or connection of lecturers schedules [Ouyang and Zhu 2007].

Another approach to recognize relationships is to use external taxonomies ([Matos-Junior et al. 2012]) or to build an architecture with hierarchies to organize objects in levels, so that these relationships among the objects become the relationships between the levels ([Sathiyamurthy et al. 2012]).

We do not assume that authors of educational material create metadata, but absence of metadata complicates the use of techniques that need this information. Therefore, we will use an approach similar to Explicit Semantic Analysis (ESA) presented in [Gabrilovich and Markovitch 2007]. The latter used a list of concepts to relate texts with Wikipedia articles. As will be seen in our case studies, we relate educational materials using text extracted from these materials, articles from Wikipedia and a taxonomy from an external authoritative source.

2.5. Analysis using graph databases

We can characterize a graph database through its data model that differentiates it from traditional relational databases [Angles and Gutierrez 2008]. A data model is a set of conceptual tools to manage and represent data, consisting of three components [Codd 1980] : 1) data structure types, 2) collection of operators or inferencing rules, and 3) a collection of general integrity rules. Data in a graph database are stored and represented as nodes, edges, and properties.

Each graph database management system has its own specialized graph query language, and there are many graph models. For example, many graph databases based on Resource Description Framework (RDF) use SPARQL[6] (SPARQL Protocol and RDF Query Language), but Neo4J [7], a graph database widely used in research, uses the Cypher language. Finally, integrity rules in a graph database are based on its graph constraints. Several researchers have adopted graph representations and graph database systems as a computational means to deal with situations where relationships are first-class citizens (e.g. [Cavoto et al. 2015]). They interpret scientific data using concepts of linked data, interactions with other data and topological properties about data organization.

As reported by Khan et al. [Khan et al. 2012], a graph database can handle directly a wide range of queries such as those expected in our work and which would otherwise require deep join operations in normalized relational tables. Cavoto et al. [Cavoto et al. 2015] argues that for analysis of data focusing on a network, complex connections or objects and their interactions, it is better to use graph databases than the relational model, considering it is usually necessary to create complex and/or inefficient SQL queries to derive the relationships.

Trying to solve the problem of finding similarities, Gater et al. [Gater et al. 2011] represented process models as graphs to reduce the problem of process matching to a

[6]https://www.w3.org/TR/rdf-sparql-query/

[7]http://neo4j.com/

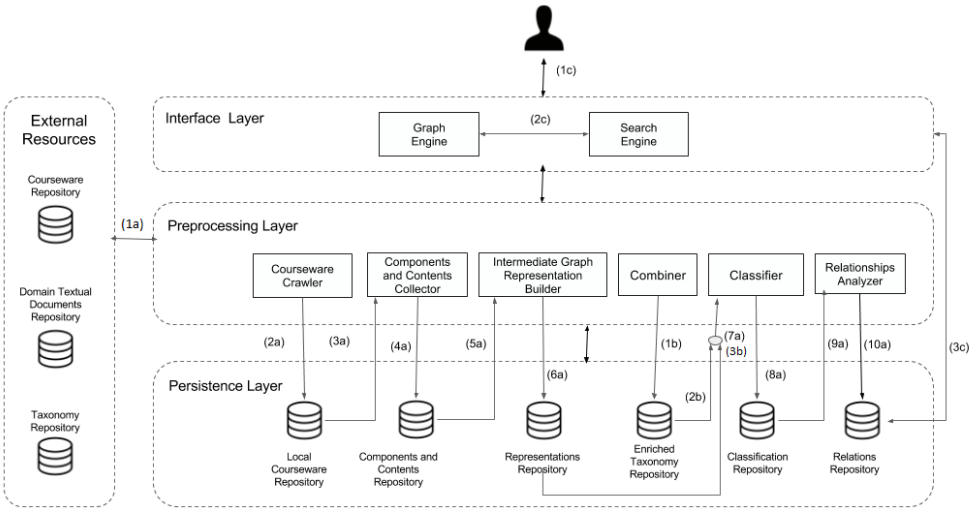


Figure 1. System Architecture for Analysis of Relationships among Educational Material Contents.

graph matching problem. Our research is inspired by the same concepts. We use graph databases to store relationships to take advantage of pattern matching algorithms. Also, using a graph database will help to analyze relations among content, to compare and check the similarities between lessons and lecturers. Algorithms such as Minimax, Betweenness Centrality and Clique may be used and thus facilitate the analysis of the topics extracted from educational materials.

There are many kinds of graph data structures. We chose to model data via *property graphs*, because this allows to create descriptive properties attached to nodes and edges. In our case, the nodes will be the educational materials, and the properties inserted into the edges will describe the relationships among the contents of the nodes. As far as we know, this is the first proposal to use graph databases with information about relationships among contents of educational materials connected to edges.

2.6. Integration of multimedia data

Work that performs the integration of multimedia data from various sources usually focus in one kind of multimedia data, e.g. web pages, ([Mishra et al. 2010, Silva and Santanchè 2009]) and/or exploit metadata to fusion multiple data about the same real-world object in a single database record ([Santanchè et al. 2014, Beneventano et al. 2011]). Examples of metadata used are: author's name, file creation date, labels.

In these proposals, search is performed among different media by searching the metadata describing the stored objects. It is also necessary to implement various different functions to perform similarity search. In our research, we do not consider metadata; rather, we seek to use the contents of educational material and external sources to integrate multimedia data.

3. The CIMAL's Architecture

CIMAL's architecture is a novel design to support the analysis of relationships among educational material based on their implicit topics. This architecture combines multiple

algorithms for content extraction and classification of topics given a suite of educational material repositories.

Figure 1 presents an overview of our architecture, which comprises three layers. The *Persistence Layer* is composed by six repositories: *Local Courseware, Components and Contents, Representations, Enriched Taxonomy, Classification and Relations*. The *Preprocessing Layer* prepares data from educational material for subsequent search. The latter provides all the services needed to look for materials using graph algorithms. These services can be accessed through the *User Interface* by lecturers and students.

The first step is to set up the repositories (actions represented by arrows with letters 'a' and 'b') before users can perform a search (arrows with letter 'c') . Preprocessing starts when the *Courseware Crawler* imports such materials from external resources (1a) and stores them in a *Local Courseware Repository* (2a). Next, the *Components and Contents Collector* extracts texts and the position of these texts from the materials in the Local Courseware Repository (3a). Extracted data are stored in the *Components and Contents Repository* (4a). Next, the *Intermediate Graph Representation Builder* creates a graph representation for each material from the repositories via the components and contents stored by the previous step (5a). These representations are stored in the *Representations Repository* (6a).

In parallel, the *Combiner*, also proposed in our research, imports an external taxonomy from a *Taxonomy Repository*, and a set of external expert texts from *Domain textual documents Repository* (1a). These data are unified in an Enhanced Taxonomy, in which each concept of the taxonomy has a reference to a text by experts, and stored in the *Enriched Taxonomy Repository* (1b).

Once representations and enriched taxonomy repositories are created, the *Classifier* is ready to define the topics covered in each of the materials (2b,3b,7a). This information is then stored in the *Classification Repository* (8a).

Lastly, the *Relationships Analyzer* looks for prespecified relationships among the items and their topics in the Classification Repository (9a), creating the *Relations Repository* (10a).

All preprocessing steps must be performed every time we add educational material, taxonomy or texts from a domain textual base.

After such preprocessing, lecturers and students can run queries through the *Interface Layer* (1c). It redirects the query to the *Graph Engine* and the *Search Engine* (2c). The latter accesses the *Relations Repository* (3c) to find relevant educational materials that are related to the user query.

4. Implementation

The CIMAL software is the first implementation of the architecture described in Section 3. We have developed the components of Interface and Preprocessing Layer using JAVA code, our texts come fromWikipedia, the taxonomy from ACM Computing Classification System[8], and methods of Apache Lucene[9], a high-performance full-featured text search

[8]https://www.acm.org/publications/class-2012

[9]https://lucene.apache.org/

engine library.

Since CIMAL uses graphs to perform relationships analysis, the Persistence Layer stores all data in a database with native support for graphs (Neo4j[10]). With this approach, we are able to use already established technologies and solutions for processing graphs. We chose the Neo4j database system because it is the most popular graph database in big companies (e.g. eBay and Wallmart) and in research, according to the Db-Engines site[11], an initiative to collect and present information on 341 database management systems.

Our main implementation is divided in four steps: (Step A) Extraction of elements of interest; (Step B) Intermediate Representation Instantiation – based on the schema defined in our research; (Step C) Intermediate Representation Analysis; (Step D) Interaction with users.

4.1. Step A - Extraction of elements of interest

At Step A, the Components and Contents Collector extracts components from material based on a Java Framework called DDEx[12] and several APIs for document handling. It scans educational material based on a set of positional rules defined by users and identifies the desired components. Each identified component is encapsulated in a standard representation and forwarded to Step B.

The following is an example of Step A applied to a file in slide format and to another in video format. Figures 2 and 3 show the components and texts, respectively highlighted through ellipses and rectangles, that will be used for classification.

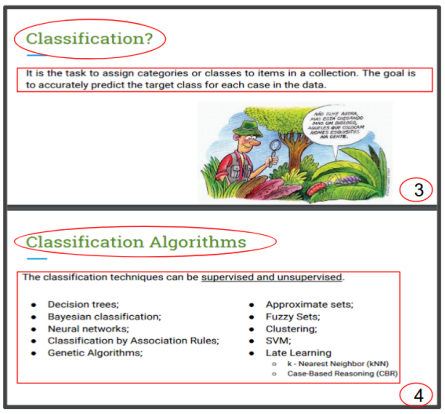


Figure 2. Components and text extracted from slides.

The texts from header and body, and number of slides were extracted automatically using DDEX as components of each slide. In addition, the texts present on the body of slides were also extracted.

Through the subtitle file, available for each of the videos, the texts and the time stamps of each of the lecturers' statements were extracted. The bold words in the figure represent the terms that were most frequent in the observed time interval.

[10]https://neo4j.com/

[11]http://db-engines.com/en/ranking/graph+dbms

[12]Open Source Project available at http://code.google.com/p/ddex

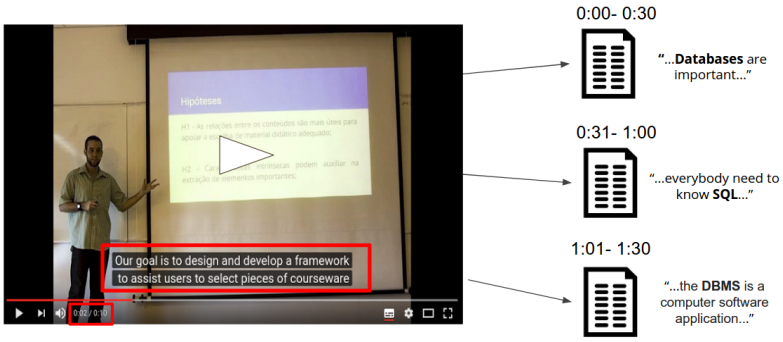


Figure 3. Components and text extracted from video subtitles.

4.2. Step B - Intermediate Representation Instantiation

Step B creates the Intermediate Graph Representation adapting the concept of shadows [Mota and Medeiros 2013] and stores this representation in a repository. The use of shadows enables the manipulation of parts of educational material without interfering with the material themselves. In the original work, shadows were implemented using XML files, but in our research we implement shadows in a graph format by the reasons already explained in Section 2.5.

The components and contents of a material are transformed into a graph where the nodes represent the elements of interest that are used in our work. These elements differ according to the kind of material, for example in a video we would like to extract the subtitles and in a slide we extract sections.

4.3. Step C - Intermediate Representation Analysis

Step C has three software modules we implemented: The first module (”Combiner” tool) is concerned with creation and storage of an enriched taxonomy. The second (Classifier tool) recognizes the topics of each Intermediate Representation according to the taxonomy and creates a document about the ”Classification of Representations”. In our studies, we defined that the words present in the components of the slides or that are among the five most repeated in videos subtitles should be 3 times more important in the classification than the words in the rest of the documents. The third module (Relationship Analyzer tool) concerns the production of information about relations, based on the ”Classification of Representations”. We developed all these tools using Java code and Apache Lucene to search documents based on text similarity.

The Combiner tool adds one page of Wikipedia to each node of the Taxonomy, thus producing an Enriched Taxonomy. Next, the Classifier tool calculates the similarity of each text of Intermediate Graph Representation (related a each educational material) for each pages of the Enriched Taxonomy.

4.4. Step D - Interaction with users

At last, in Step D users can perform queries to find relevant content. Here we implemented in Java and 2graph[13] the Interface layer tools. 2graph is a java-based API to perform Extract, Transform and Load (ETL) resources to graph structures/databases, to handle the information produced by CIMAL and interact with users.

[13] Available at http://www.lis.ic.unicamp.br/ matheus/projects/2graph

5. Research Challenges

To achieve the objective of this research the following obstacles have been faced:

1) Although widespread, the idea of sharing teaching materials still faces resistance from lecturers. In order to perform classification tests and also to verify relationships between the topics, it is necessary to find different materials but with similar approaches to explain topics. The solution found was to use materials from the same repository (Coursera) and from the Computing area, in which the idea of electronic sharing is more popular.

2) Most of the lesson videos are produced for a specific audience. Consequently, many lectures only explain concepts in a specific language, and do not produce subtitles for other audiences. Automatic transcription of captions is still a research problem. Therefore, we have selected only videos that had their subtitle produced manually, which drastically reduced the amount of educational videos available in educational repositories that could be used. Thus, we used videos from the Coursera platform, which follow a standard of subtitle production, thereby making the analysis of video content more adequate.

3) The use of graphs for analysis of relationships is very common in many research domains, but this practice is not yet widespread in the educational field. In our work we only use volunteers with knowledge in graphs to analyze the contributions of this research.

6. Case Studies

6.1. Analysis of important topics in a Specialization Course from Coursera

Coursera is a web platform that provides universal access to educational material and courses online from universities and organizations around the world. However like other producers of educational material, Coursera often does not indicate all the topics covered in a given content. This hampers distinguishing among courses.

We collected 97 sets of slides and 97 videos from the Specialization course in Data Science, offered by Johns Hopkins University [14], to be used as a case study. For this study, our enriched taxonomy was based on ACM Computing Classification System.

Using our system, we are able to discover the topics covered throughout the specialization course without requiring annotations or other extra tasks for teachers. These topics can then be briefly presented as requirements or even in a short course that would be offered to all students before enrolling in the specialization course.

We point out that CIMAL can thus also be used by lecturers to annotate and classify their materials. More details on this case study can be found at [Saraiva and Medeiros 2017].

6.2. Proposed new multidisciplinary activities in an educational institution

A second case study was conducted at an educational institution in the state of São Paulo, Brazil. We show how we find similarities among different courses, thereby highlighting possible intersections, thus revealing potential multi-course activities.

[14] https://www.coursera.org/specializations/jhu-data-science

This educational institution seeks to promote interdisciplinary activities to prepare students for the increasingly complex labor market, which requires diversity of knowledge. However, there are many courses that make it difficult to see their relationships.

Using our architecture, we were able to extract the contents and topics covered in each of the documents that regulated the courses of this institution and relate each of their contents through graphs. Documents with many relations revealed possible interactions between their respective courses.

The results of this case study were presented to the faculty of the Institute, who through a questionnaire evaluated if the information obtained could be used to elaborate activities involving courses. In total, 20 lecturers from different courses answered the questionnaire, and 75% answered that it was possible to use the information obtained to propose new interdisciplinary activities between courses.

6.3. Standardizing validation

To finalize our study, we designed a questionnaire to evaluate the classification of topics extracted from 6 materials (randomly chosen for the questionnaire does not get too long) from the ”Python for Everybody Specialization”, provided by University of Michigan. Thirty volunteers of different levels of education and specialties in sub-areas of Computer Science (2 undergraduate student, 3 undergraduate degree, 3 specialists, 6 Master in progress, 4 Master's degree, 8 PhD in progress, 4 PhD completed) gave opinions for each of five topics extracted using the CIMAL implementation. Since the course was about ”Python programming language” and in the ACM taxonomy these terms are not present, we added manually in our database the Wikipedia page about this topic.

We analyze 900 answers, in each of them a volunteer indicated if he had knowledge about the topic that is being asked. Only answers from volunteers who reported having knowledge about the topic were considered (747 answers). After this activity, we can see that CIMAL classifies the materials using pertinent topics, since 64% of the topics indicated by the framework were evaluated ”Some related (16,5%)”, ”Related (15%)” or ”Closely related (32,5%)” by the volunteers.

7. Conclusions and Future Work

This paper presented the design and implementation of CIMAL, which allows searching content from educational material, and eliciting relationships among topics. This framework contributes to helping lecturers and students navigate through collections of materials. Our implementation is validated on slides and videos from case studies and showed that the components on slides and videos can be used to classify text and relate topic of these materials.

One particular question is of interest to us: ”Can the history of courses taken by students influence the topics that the students are looking for in educational material repositories?”

To answer this question, it is necessary to collect data of user accesses to these materials. For example, data on the last courses that a student held in Coursera could be used to construct a personalized study guide on subjects that would be interesting for this student; the recommendation system could also recommend more Coursera courses.

8. Acknowledgment

We thank Prof. R. D. Peng from Johns Hopkins University and Prof. C. R. Severance, who made slides, videos and subtitles available to be used in a case study. Also we would like to thank the faculty members of Federal Institute of Education, Science and Technology of São Paulo - Campus Hortolândia, who participated in the evaluation and analysis of the results. Also we thank all the members of the Laboratory of Information System - Institute of Computing/Unicamp in which this research was conducted, who assisted several times with suggestions.

Our work partially financed by FAPESP/Cepid in Computational Engineering and Sciences (2013/08293-7), INCT in Web Science (CNPq 557.128/2009-9), and individual grants from CAPES and CNPq.

References

Angles, R. and Gutierrez, C. (2008). Survey of graph database models. *ACM Comput. Surv.*, 40(1):1:1–1:39.

Beneventano, D., Gennaro, C., Bergamaschi, S., and Rabitti, F. (2011). A mediator-based approach for integrating heterogeneous multimedia sources. *Multimedia Tools and Applications*, 62(2):427–450.

Blei, D. M. (2012). Probabilistic topic models. *Commun. ACM*, 55(4):77–84.

Cavoto, P., Cardoso, V., Vignes Lebbe, R., and Santanchè, A. (2015). FishGraph: A Network-Driven Data Analysis. In *11th IEEE Int. Conf. on eScience*, Germany.

Changuel, S., Labroche, N., and Bouchon-Meunier, B. (2015). Resources sequencing using automatic prerequisite–outcome annotation. *ACM Trans. Intell. Syst. Technol.*, 6(1):pages 6:1–6:30.

Codd, E. F. (1980). Data models in database management. *SIGPLAN Not.*, 16(1):112–114.

Egozi, O., Markovitch, S., and Gabrilovich, E. (2011). Concept-based information retrieval using explicit semantic analysis. *ACM Trans. Inf. Syst.*, 29(2):8:1–8:34.

Gabrilovich, E. and Markovitch, S. (2007). Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *IJCAI*, pages 1606–1611, CA, USA. Morgan Kaufmann Publishers Inc.

Gabrilovich, E. and Markovitch, S. (2009). Wikipedia-based semantic interpretation for natural language processing. *J. Artif. Int. Res.*, 34(1):443–498.

Gater, A., Grigori, D., and Bouzeghoub, M. (2011). A graph-based approach for semantic process model discovery. *Graph Data Management*, pages 438–462.

Jiang, J. (2012). Information extraction from text. In Aggarwal, C. C. and Zhai, C., editors, *Mining Text Data*, pages 11–41. Springer US.

Khan, A., Wu, Y., and Yan, X. (2012). Emerging graph queries in linked data. In *ICDE*, pages 1218–1221. IEEE.

Learning Technology Standards Committee of the IEEE (2002). Draft standard for learning technology - learning object metadata. Technical report, IEEE Standards Department, New York.

Little, S., Ferguson, R., and Rüger, S. (2012). Finding and reusing learning materials with multimedia similarity search and social networks. *Technology, Pedagogy and Education*, 21(2):pages 255–271.

Matos-Junior, O., Ziviani, N., Botelho, F. C., Cristo, M., Lacerda, A., and da Silva, A. S. (2012). Using taxonomies for product recommendation. *JIDM*, 3(2):pages 85–100.

Mishra, S., Gorai, A., Oberoi, T., and Ghosh, H. (2010). Efficient Visualization of Content and Contextual Information of an Online Multimedia Digital Library for Effective Browsing. *WI-IAT2010*, pages 257–260.

Mota, M. S. and Medeiros, C. B. (2013). Introducing shadows: Flexible document representation and annotation on the web. *ICDE Workshops*, pages 13–18.

Ouyang, Y. and Zhu, M. (2007). eLORM: Learning object relationship mining based repository. *Proceedings - IEEE Int. Conf. on E-Commerce Technology and CEC/EEE*, pages 691–698.

Pereira, B. (2014). Entity Linking with Multiple Knowledge Bases: An Ontology Modularization Approach. In *ISWC*, pages 513–520. Springer.

Ricarte, I. L. M. and Junior, G. R. F. (2011). A methodology for mining data from computer-supported learning environments. *Informática na educação: teoria & prática*, 14(2).

Romero, C. and Ventura, S. (2013). Data mining in education. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 3(1):12–27.

Rossi, R. G., Rezende, S. O., and Lopes, A. A. (2015). Term network approach for transductive classification. volume 9042, pages 497–515. Springer International Publishing.

Santanchè, A., Longo, J. S. C., Jomier, G., Zam, M., and Medeiros, C. B. (2014). Multi-focus research and geospatial data - anthropocentric concerns. *JIDM*, 5(2):pages 146–160.

Saraiva, M. C. and Medeiros, C. B. (2016). Use of graphs and taxonomic classifications to analyze content relationships among courseware. In *Brazilian Symposium on Databases, SBBD 2016, Salvador, Bahia, Brazil*, pages 265–270.

Saraiva, M. C. and Medeiros, C. B. (2017). Finding out topics in educational materials using their components. In *47th Annual IEEE Frontiers in Education Conference (FIE), Indianapolis, IN, USA, pp. 1-7*.

Sathiyamurthy, K., Geetha, T. V., and Senthilvelan, M. (2012). An approach towards dynamic assembling of learning objects. In *ICACCI*, pages 1193–1198. ACM.

Silva, L. M. D. and Santanchè, A. (2009). ARARA: Autoria de Objetos Digitais Complexos Baseada em Documentos. *Simpósio Brasileiro de Informática na Educação*, (2009):10.

Zhuang, Y. (2017). Bag-of-discriminative-words (bodw) representation via topic modeling. *IEEE Transactions on Knowledge and Data Engineering*, 29(5):977–990.

Caracterização e Comparação das Campanhas do Outubro Rosa e Novembro Azul no Twitter

Roberto Walter[1], **Karin Becker**[1]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

roberto.wtr@gmail.com, karin.becker@inf.ufrgs.br

Abstract. *The Pink October and Blue November campaigns seek to spread education and awareness regarding breast and prostate cancers. Given the success of the Pink October campaigns and the low male engagement in the Blue November, in this article, we present a pioneer comparative evaluation of these campaigns in the online context. Considering Twitter as a platform that provides community engagement, we study the demographic characterization of gender and age, geographic location, activity periods, and user engagement using tweets from the 2017 Pink October and Blue November campaigns. We have identified that these campaigns have reached their target audience, i.e. men and women over 40 years of age. We also discovered patterns in the activities throughout the respective period within campaigns, as well as differences when different countries are considered. Pink October's outreach is definitely greater, measured in terms of the number of tweets, re-tweets, as well as the ability to spread through the social network*

1. Introdução

Estudos estimam que uma em cada oito mulheres irá desenvolver o câncer de mama durante sua vida [Altekruse et al. 2010]. A realização de exames de prevenção é fundamental para reverter este quadro. A campanha do Outubro Rosa[1] tem anualmente focado em aumentar a participação feminina nestes exames, bem como educar as pessoas e aumentar os cuidados em geral referentes ao câncer de mama [Jacobsen and Jacobsen 2011, National Breast Cancer Foundation 2017]. Esta campanha organiza caminhadas, eventos esportivos, e distribui materiais de comunicação e vestimentas rosas. O principal objetivo é motivar mulheres a realizar exames que detectam o câncer em estágios ainda iniciais, bem como angariar recursos financeiros. Estes esforços têm obtido muito êxito. Por exemplo, nos EUA o número de mulheres que realizaram exames preventivos subiu de 26% em 1987, para aproximadamente 72,4% em 2010 [for Disease Control and Prevention 2011]. Outro estudo [Glynn et al. 2011] reporta que no mês de outubro há mais interesse no tópico de câncer de mama em pesquisas *online*. No entanto, este mesmo nível de consciência não é observado para outros tipos de câncer, como o de próstata e o de pulmão.

Aproximadamente 63% dos pacientes de câncer buscam informações oncológicas na *internet* [Castleton et al. 2011], e alguns trabalhos têm se dedicado a examinar a importância do uso de mídias sociais, como o Twitter, para propagar o conhecimento sobre a saúde pública [Himelboim and Han 2014,

[1]O Outubro Rosa teve início ainda nos primeiros anos de 1990, quando outubro foi reconhecido oficialmente pelo governo dos EUA como o mês de consciência sobre o câncer de mama.

Laranjo et al. 2014, Bravo and Hoffman-Goetz 2016]. O Twitter tem sido utilizado em campanhas *online* de diferentes propósitos, como no movimento Black Lives Matter por equidade racial [Olteanu et al. 2016], contra violência baseada no gênero [ElSherief et al. 2017], primavera árabe [Lotan et al. 2011], e também nas campanhas do Outubro Rosa [Thackeray et al. 2013, Nastasi et al. 2017] e Novembro Azul[2] [Bravo and Hoffman-Goetz 2016, Prasetyo et al. 2015]. As campanhas do Outubro Rosa e Novembro Azul (na sequência, referenciadas como OR e NA respectivamente) são estimuladas no Twitter através do uso de *hashtags* como *#breastcancerawareness* e *#cancerdemama* para o OR, e *#Movember* e *#NovembroAzul* para o NA. Estas campanhas aumentaram o conhecimento da população permitindo um trabalho mais efetivo de prevenção, exames, conhecimento sobre tratamentos, pesquisas e posições políticas [Edge 2006]. Porém, divergências nos padrões de desenvolvimento dessas campanhas podem ser fatores que contribuem para os baixos índices de engajamento dos homens em exames de detecção do câncer de próstata nos seus estágios iniciais.

Alguns trabalhos foram desenvolvidos com o intuito de apresentar indicadores sobre períodos de atividades, engajamento, conteúdo e caracterização dos usuários da campanha do OR [Thackeray et al. 2013, Nastasi et al. 2017], ou comparação de conteúdo do Twitter referentes a diferentes tipos de câncer [Borgmann et al. 2016]. No entanto, não há uma avaliação de atividade durante o mês de campanha, de caracterização e localização dos perfis, e do engajamento dos usuários para a campanha do NA. Também não foram identificadas abordagens comparativas entre estas campanhas, apesar de seu propósito similar. A comparação das propriedades de ambas campanhas pode permitir compreender fatores que explicam o sucesso do OR, que expandiu os seus números desde o início das campanhas, e a participação ainda tímida no NA. Desta forma, é importante averiguar se existem divergências nos padrões de desenvolvimento das campanhas, e a partir disso identificar se o público alvo do NA está sendo abordado da forma adequada como está sendo o público alvo do OR. Neste contexto de campanhas convencionais e *online*, será realizada uma avaliação e comparação das campanhas *online*.

O objetivo deste trabalho é permitir um melhor conhecimento sobre as campanhas *online* do NA, além de caracterizar e compará-las às campanhas do OR. Nossa pesquisa busca entender e comparar o engajamento dos usuários proporcionado através de *tweets*, suas características demográficas, e os períodos de mais atividades em cada campanha. Para isso, é avaliada uma base de dados com aproximadamente 680,000 *tweets*, coletados entre setembro e dezembro de 2017. Em nossa análise buscamos responder as seguintes perguntas sobre o NA, comparando-o com o OR:

- **QP 1**: Os usuários envolvidos nas campanhas apresentam características de perfil demográfico e geográfico similares?
- **QP 2**: As campanhas apresentam características temporais similares?
- **QP 3**: As campanhas apresentam abrangência similar na rede social?

Este estudo constitui uma experiência pioneira comparando campanhas de câncer no Twitter, com as seguintes contribuições:

- Complementamos estudos anteriores (e.g. [Nastasi et al. 2017, Jacobson and Mascaro 2016][Thackeray et al. 2013][Prasetyo et al. 2015] com

[2]O Novembro Azul surgiu na Austrália, em 2003, chamado Movember, a aproveitando as comemorações do Dia Mundial de Combate ao Câncer de Próstata, realizado a 17 de novembro.

uma avaliação comparativa entre OR e NA, e análises adicionais: i) o público envolvido em cada campanha, verificando se corresponde ao respectivo público-alvo; ii) os padrões temporais de atividades em cada campanha, detectando diferenças/semelhanças por campanha e país; iii) diferenças na cobertura de tweets em cada campanha, mostrando que o NA tem alcance limitado e que organizações e celebridades não desempenham um papel proeminente. Essas análises podem ser estendidas a outras campanhas sobre câncer.
- Geramos métricas para uma campanha com resultados positivos (OR) e a contrastamos com uma campanha similar (NA) com menos engajamento. Essa comparação ajuda a entender os fatores que influenciam o alcance da campanha do NA no Twitter e, consequentemente, aumenta o engajamento da população-alvo em exames de detecção precoce do câncer.
- Estabelecemos um *baseline* para essas campanhas, o que nos permite monitorar sua evolução no ano futuro.

Este artigo está organizado da seguinte forma: na próxima seção são apresentados trabalhos relacionados e as técnicas utilizadas neste trabalho. A Seção 3 define os dados, métodos e proposta deste artigo. A Seção 4 expõem os experimentos e a Seção 5 apresenta as conclusões e trabalhos futuros.

2. Trabalhos Relacionados

Vários são os trabalhos que estudam o perfil de engajamento de usuários do Twitter em diferentes causas (e.g. equidade racial, violência baseada no gênero, câncer). A Tabela 1 lista alguns trabalhos com seu contexto de aplicação, destacando desafios comuns a todos: coleta e integração de dados, avaliação de dados temporais, caracterização de usuários por categorias, engajamento de usuários nas campanhas, determinação de informações demográficas, incluindo geolocalização.

Estes trabalhos buscam primariamente criar um *dataset* válido e representativo do contexto estudado. A maioria usa o Twiter, restringindo a tweets que contenham palavras chaves ou *hashtags* em um período determinado, usando a API de integração do Twitter [3] ou outras ferramentas integradas na API. As análises temporais em geral são avaliações de volume de *tweets* e ativismo dos usuários durante períodos de campanhas *online*. O engajamento dos usuários geralmente é medido pela atividade de postagem de *tweets* e *retweets*, detalhado no contexto de informações demográficas como sexo e idade, geolocalização ou categorização de usuários. Assim, busca-se definir as características de participantes e de perfis que possam ter mais ou menos influência/participação nas mobilizações estudadas. Contudo, essas definições demográficas e geográficas são um problema. Por um lado, ao capturar o perfil de um usuário, o Twitter não solicita informações de sexo e idade, e permite que a localização geográfica seja informada em um campo aberto, que muitas vezes é inválida. Portanto, alguns recursos como o Face++[4], Google Maps, ou plataformas coletivas de definição de dados têm sido utilizadas para buscar uma definição destes dados do usuário com informações que o mesmo possa ter informado em seu perfil, como por exemplo a imagem de perfil ou localização. O Twitter também permite que o usuário geo-referencie os seus *tweets*, mas o número de tweets geo-referenciados é menor que 2% [Schulz et al. 2015]. Assim, utiliza-se frequentemente

[3]https://developer.twitter.com

[4]https://www.faceplusplus.com/

Tabela 1. Visão geral de dados e métodos de trabalhos relacionados.

Obra	Integração Dados	Contexto	Análise Temporal	Engajamento	Tipo de Usuário	Demografia			Geolocalização	
						Idade	Sexo	Técnica Definição	Estado Político	Técnica Definição
ElSherief et al. 2017	API Twitter	Violência baseada no gênero	-	Sim	-	Sim	Sim	Face++	-	-
De Choudhury et al. 2016	API Twitter	Equidade Racial	Sim	-	-	-	-	-	Estado	Auto-relatado pelo usuário e filtro em Nominatim library
Olteanu et al. 2016	API Twitter	Equidade Racial	Sim	-	Sim	Sim	Sim	Crowdworkers (Crowdflower)	-	-
Lotan et al. 2011	API Twitter	Primavera Árabe	Sim	Sim	Sim	-	-	-	País	Auto-relatado pelo usuário
Glynn et al. 2011	Google Insights for Search	Outubro Rosa	Sim	-	-	-	-	-	-	-
Thackeray et al. 2013	API Twitter	Outubro Rosa	Sim	Sim	Sim	-	-	-	-	-
Borgmann et al. 2016	Symplur, Tweet Archivist, Twitonomy	Oncologia Urológica	Sim	Sim	Sim	-	-	-	País	Twitonomy
Nastasi et al. 2017	Symplur	Outubro Rosa	-	Sim	Sim	-	-	-	País/Continente	-
Prasetyo et al. 2015	API Twitter	Novembro Azul	-	Sim	Sim	-	-	-	País/Continente	Auto-relatado pelo usuário
Jacobson and Mascaro 2016	API Twitter	Novembro Azul	Sim	Sim	-	-	Sim	-	-	-

as informações disponíveis no perfil, ainda que a informação possa ser imprecisa (e.g. o usuário está em local diferente do relatado no perfil, quando da postagem).

Campanhas relacionadas ao câncer no Twitter aumentaram a conscientização da população sobre o câncer, prevenção e tratamentos [Edge 2006]. Diferenças em tópicos de tweets de acordo com o tipo de câncer foram examinadas em [Borgmann et al. 2016]. As campanhas OR/NA foram caracterizadas em termos de atividades, nível de engajamento, e categorias de usuários [Thackeray et al. 2013, Nastasi et al. 2017]. Uma análise das atividades de angariação de fundos relacionadas com NA em diferentes países foi desenvolvida em [Prasetyo et al. 2015]. O presente trabalho complementa estes trabalhos relacionados, desenvolvendo uma análise comparativa entre as campanhas OR e NA, a fim de auxiliar na compreensão de fatores que contribuem a um maior engajamento da população-alvo nas campanhas on-line.

3. Dados e Métodos

Neste trabalho, analisamos as características temporais das atividade das duas campanhas de câncer, associadas as respectivas informações demográficas e de localização dos usuários e seu engajamento, bem como o alcance das campanhas.

Os dados de *tweets* e perfil dos usuários foram coletados do Twitter a partir da API de consulta aos dados públicos. A consulta foi efetuada com um conjunto inicial de *hashtags* que foram levantadas pelos autores como relacionadas às campanhas do OR e NA nos anos anteriores, e ao examinar as relações no Twitter, agregou-se um novo conjunto de *hashtags* resultando no conjunto da Tabela 2.

As *hashtags* contemplam as principais campanhas nos idiomas inglês, português, espanhol e alemão. A definição do período foi pelo mês da campanha, precedido e sucedido de uma semana, o que resultou no período de 24 de setembro de 2017 a 07 de

Idioma	OR	NA
Português	#OutubroRosa, #Outubrorosa2017, #PrevençãoContraoCâncerDeMama, #outubrorosabr, #cancerdemama	#CâncerDePróstata, #NovembroAzul, #NovembroAzul2017
Inglês	#breastcancerawareness, #thinkpink, #pinkoctober, #walkagainstcancer, #breastcancer, #IDriveFor, @AmericanCancer, #breastcancerawarenessmonth, #projectpinkblue, #raceforthecure, #BCSM, #BRCA, #pinktober, #chokecancer	#Movember, #ProstateCancer, #BlueNovember, #beatcancer
Espanhol	#OctubreRosa, #miluchaesrosa, #luchacontraelcancerdemama	#noviembreazul
Alemão	#RosaOktober, #bröstcancer	#Prostatakrebs

Tabela 2. Conjunto de *hashtags*

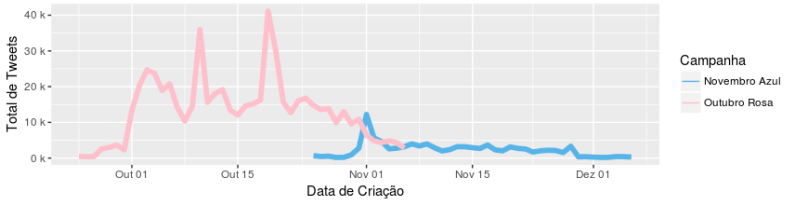


Figura 1. Distribuição de Tweets por Data de Criação

dezembro de 2017. No total foram coletados 678.968 *tweets* referentes a ambas campanhas, postados por 213.905 usuários diferentes. A distribuição do número de postagens por data é mostrada na Figura 1.

A fim de avaliar a abrangência das campanhas, foram mantidos tanto os *tweets* originais, quanto os *retweets*. Após a obtenção dos dados de *tweets* e usuários do Twitter, aplicamos o *Face++* para definir a idade e gênero dos usuários a partir de suas fotos do perfil. Essa escolha foi motivada pela acurácia mínima do Face++ conforme reportado em [Fan et al. 2014]. Aproximadamente 44% do nosso *dataset* não teve uma definição demográfica pelo Face++. A API do Google Maps[5] foi utilizada para obtenção do país do usuário a partir da informação de localidade que o usuário informa em um campo aberto no seu perfil no Twitter (apenas 0.33% dos *tweets* coletados são georreferenciados).

Além disto, foi criado um atributo nas informações do usuário que indica a sua categoria, onde ele pode ser classificado como Celebridade, Organização, ou Individuo. Para o perfil se enquadrar como Celebridade, da mesma forma como definido em [Thackeray et al. 2013], o perfil deve ser verificado como verdadeiro pelo Twitter, possuir mais de 100 mil seguidores, e um gênero (masculino ou feminino) identificado pelo Face++. O usuário da categoria Organização é um perfil verificado pelo Twitter e que não possui o gênero identificado pelo Face++. Os demais perfis são classificados como Individuo ou Desconnhecido.

4. Experimentos

Nesta seção são apresentados os experimentos para as questões de pesquisa (QP) definidas anteriormente, que buscam atingir os objetivos definidos na seção anterior.

4.1. QP 1: Os usuários envolvidos nas campanhas apresentam características de perfil demográfico e geográfico similares?

Na Figura 2 pode-se visualizar a distribuição por gênero e campanha, em termos de percentagem do total de tweets. Identifica-se que cada gênero engaja mais na campanha da qual é alvo, isto é, mulheres na OR e homens no NA. Contudo, observa-se também que nas duas campanhas o gênero que mais engaja é o masculino, representando 62.7% dos usuários engajados no NA. Mesmo na campanha OR, o gênero masculino representa o grupo com maior envolvimento (49.09%). Há também um grupo menor, representado pelas organizações, que possuem participação equilibrada em ambas campanhas, com 7.26% no NA e 5.63% no OR.

No que diz respeito à participação das faixas etárias, os usuários foram separados em um grupo com até 40 anos (-40), e acima de 40 anos (41+). Esses grupos foram

[5]https://developers.google.com/maps/

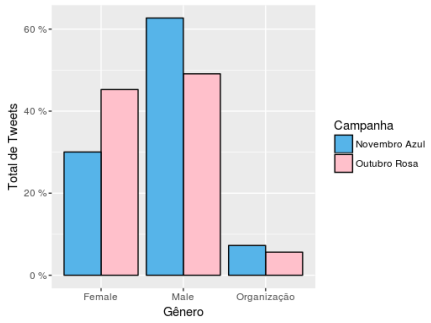


Figura 2. Distribuição por Gênero X Campanha

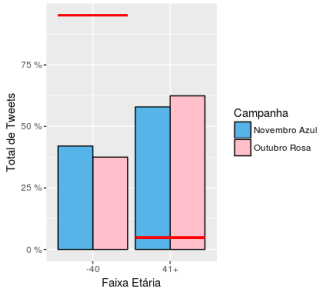


Figura 3. Distribuição por Faixa Etária x Campanha

criados para verificar se as pessoas com mais de 40 anos, que são o público alvo da realização dos exames preventivos, possuem participação similar entre as campanhas do OR e NA, e os *tweets* de propósito geral. A Figura 3 apresenta a participação dos dois grupos etários nas campanhas em termos de percentagens sobre o total de tweets. Pode-se visualizar uma maior participação do grupo etário 41+, com 62.46% no OR e 57.95% no NA. A distribuição de cada grupo nas duas campanhas é muito próxima, sendo que no OR o grupo 41+ tem uma participação levemente maior, quando comparado ao NA. Para estabelecer um *baseline*, indicamos a participação destas mesmas faixas etárias na postagem de *tweets* em geral de acordo com um estudo [Sloan et al. 2015] (95.2% e 4.8% para os grupos -40 e 41+, respectivamente). Esse alto índice de participação, comparado a *tweets* em geral, indica que ambas campanhas estão atingindo os seus públicos alvos.

As Figuras 4 e 5 apresentam a distribuição de gênero e faixa etária para os cinco países com o maior número de *tweets* sobre as campanhas. Pode-se verificar que com exceção do Canada, em todos os países a campanha do OR atrai maior participação quando comparada a do NA. Os Estados Unidos e o México são os países que apresentam as maiores diferenças, onde a participação no NA está abaixo dos 10%.

Quando detalhada por gênero (Figura 4), observamos uma maior participação de organizações no OR, com destaque para o Brasil, México e UK. No NA, o envolvimento é mínimo. O NA engaja mais sua população alvo, i.e. masculina, exceto nos Estados Unidos, onde é igualitária. Já a campanha OR tem comportamentos distintos conforme o país: maior participação feminina em UK e Canada, igualitária no Brasil, e minoritária nos Estados Unidos. Quanto à idade (Figura 5) , observa-se uma predominância de participação do público alvo (41+), exceto pelo Brasil e México, onde há um equilíbrio entre as duas populações.

Conclui-se assim que ambas as campanhas atingem seu público alvo, no tocante

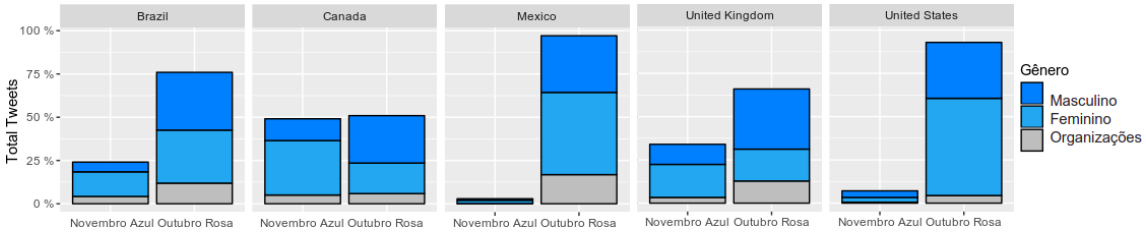


Figura 4. Distribuição de gênero por país

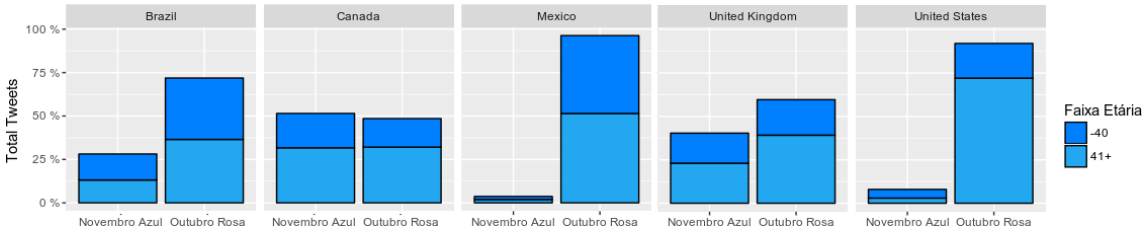


Figura 5. Distribuição de faixa etária por país

à gênero e faixa etária, mas que os níveis de consciência e engajamento são afetados pela cultura ou políticas próprias a cada país. A participação de organizações na causa do NA ainda é tímida.

4.2. QP 2: As campanhas apresentam características temporais similares?

Os números da distribuição de *tweets* por data de postagem conforme foram apresentados na Figura 1, onde se pode verificar que a quantidade de *tweets* nos dias que precedem as campanhas são relativamente baixos, comparados ao respectivo período oficial de campanha. As duas campanhas apresentaram um pico no início de seu respectivo mês. No caso do NA, após os primeiros dias a participação teve um declínio e estabilizou. A campanha do OR, por outro lado, teve três (3) picos (um deles no início da campanha), sendo o declínio da participação observado somente ao final do mês.

O padrão de semelhança entre os países quando considerado o engajamento total nas duas campanhas (Figuras 4 e 5) não é observados quando a participação é detalhada ao longo do período da campanha, como mostra a Figura 6. Canadá, Reino Unido e Estados Unidos, que apresentam proporções de participação total bastante diversos, revelam similaridade nas suas atividades detalhadas por campanha. Para o NA, observa-se um pico de *tweets* nos primeiros dias da campanha, seguida de uma posterior estabilização, enquanto que no OR, existe uma estabilidade ao longo da campanha. O Brasil apresenta um grande pico no início do NA, bem maior que o pico inicial da campanha OR, seguidos de bastante instabilidade de participação ao longo de ambas campanhas. O México também apresenta bastante instabilidade de atividade durante os períodos, mas em um padrão distinto.

Para confirmar estas semelhanças e diferenças de atividades nos países, calculamos as correlações entre as séries do OR e NA de todos pares de países pela medida de Normalized Cross-Correlation. Os resultados são apresentados na Tabela 3 para o OR e NA nas áreas rosa e azul, respectivamente. É possível averiguar que de fato o Canadá, Reino Unido e Estados Unidos apresentam similaridade entre as atividades temporais das campanhas, com correlações acima de 0.90 no NA e 0.70 no OR. Em termos das diferenças, pode-se confirmar também que o Brasil não possui correlações fortes com os demais países, e que o México apresenta valores ainda menores.

A Figura 7 apresenta o percentual acumulado dos *tweets* ao longo dos dias da campanha e confirma a maior atividade do NA no início da campanha. Pode-se verificar que no 10º dia o NA já atinge 28.45% dos *tweets* postados, enquanto que a campanha do OR atinge aproximadamente esse percentual somente no 16º dia da sua campanha. No entanto, as duas curvas apresentam regularidade no seu crescimento durante o mês. As exceções desse crescimento regular são esse crescimento acelerado já mencionado do NA no início do seu mês, um declínio do NA mais acentuado no final, e um crescimento do

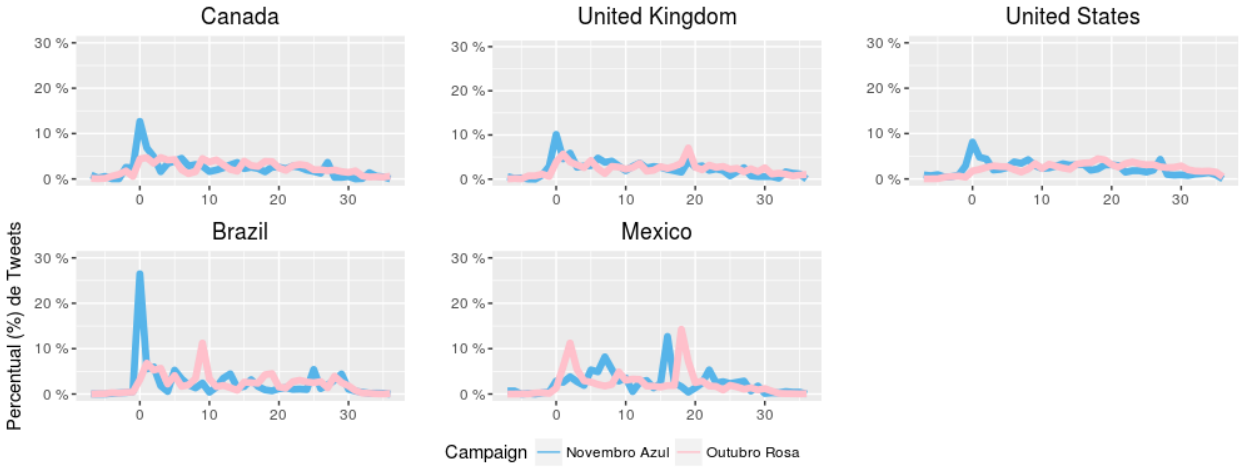


Figura 6. Distribuição Países

Tabela 3. Correlação do nível de atividade por campanha. As correlações OR e NA estão representadas nas cores rosa e azul.

	Canadá	Reino Unido	Estados Unidos	Brasil	México
Canadá		0.8131533	0.7100765	0.7614922	0.6226711
Reino Unido	0.9250876		0.7123116	0.6804556	0.6579495
Estados Unidos	0.9094715	0.9402248		0.5670101	0.551835
Brasil	0.8316607	0.7470957	0.718368		0.6280569
México	0.3545886	0.3862714	0.4534587	0.216265	

OR por volta do 25º dia.

Assim, identifica-se que alguns países possuem atividades de *tweets* similares nas campanhas, e que em outros há diferenças nestes períodos de aplicação. Em geral, a campanha do OR apresentou mais regularidade de atividades, e o NA teve oscilações de forma mais ou menos acentuada em cada um dos países.

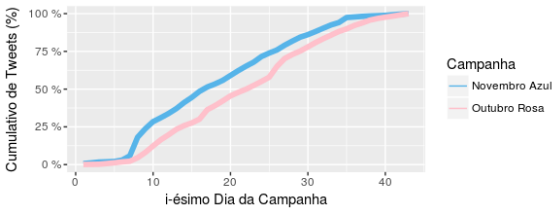


Figura 7. Percentual Acumulado (%) dos Tweets por Dia de Campanha

4.3. QP 3: As campanhas apresentam abrangência similar na rede social?

Foi realizada uma avaliação de abrangência para saber em quais campanhas os *tweets* atingem mais pessoas. Quando avaliado o número de *tweets* por categoria de usuário, em todas as categorias o número de usuários e *tweets* é maior na campanha do OR, comparado ao NA, como mostra a Tabela 4. Outro indicador é a média de *tweets* por usuário, a qual também é maior em todas as categorias para o OR. Isto indica que, além da maior participação de usuários nas campanhas, em média os usuários envolvidos na campanha do OR se envolvem com mais postagens, se comparados aos participantes no NA.

Tabela 4. Características dos *tweets* por tipo de usuário

	Organizações		Individuos		Celebridades		Total	
	OR	NA	OR	NA	OR	NA	OR	NA
Nº Usuários	2364	1007	172826	46025	257	81	175447	47113
Nº de Tweets	17016	3505	564646	96329	643	142	582305	99976
Média	7.19	3.48	3.26	2.09	2.50	1.75	3.31	2.12

Sabendo que os *tweets* podem ser retuitados e atingir uma rede maior do que apenas os seguidores imediatos do perfil que postou a mensagem, buscou-se compreender as relações por meio dos *retweets* entre os usuários. Para cada campanha, foi construído um grafo que contém a estrutura das relações de *retweets* entre os usuários, apresentados nas Figuras 8 e 9. Cada vértice do grafo representa um usuário, onde as arestas que os conectam representam os *retweets* entre os usuários. As cores dos vértices representam um tipo de estrutura, definida pelo número de usuários e de *retweets*.

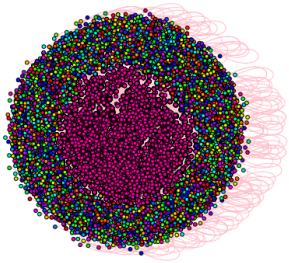


Figura 8. Conexões via *retweets* entre usuários do OR

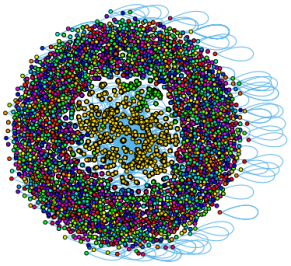


Figura 9. Conexões via *retweets* entre usuários do NA

Todos os subgrafos da rede de conexão entre os usuários, juntamente com o número de usuários (V) e de repetições no grafo (N), são apresentados nas Figuras 10 e 11, ordenados nas linhas por número de usuários em cada estrutura. Se tomarmos as primeiras linhas de cada figura, pode-se verificar que as estruturas do OR possuem um maior número de vértices, e que estas são mais utilizadas do que as maiores estruturas em vértices do NA. Isto indica que o nível de conexão entre os usuários do OR é maior do que a do NA, com os usuários compartilhando bem mais informação de um modo geral. Outro padrão interessante é o alto número de usuários centralizados que são retuitados, e de *retweets* cruzados, onde usuários retuítam-se entre si.

Pode-se reforçar estas suposições com uma apresentação da frequência de número de *retweets* por data de criação, mostrada na Figura 12 para ambas as campanhas. É verificado um maior número de *retweets* de *tweets* relacionados ao OR, onde a quantidade de *retweets* fica em torno de 300 em diversas datas, além de um número bem maior de *tweets* retuitados por dia. Por outro lado, no NA essa máxima de *retweets* fica um pouco acima de 50, ou seja, 16,6% do número do OR. Pode-se verificar também que a mediana (linha horizontal nos gráficos) é bem mais alta no OR (8 *retweets*), comparada ao NA (apenas 4). Esta superioridade também é observada em termos do 3º quartil, onde são observados 34 *retweets* para o OR, e apenas 8 para o NA.

Finalmente, a distribuição do grau de conexões a partir de *retweets* é exibida na Figura 13 para as duas campanhas, para cada uma das diferentes categorias. A maioria dos usuários são retuitados poucas vezes, enquanto apenas pouquíssimos usuários possuem várias conexões por *retweets*. Isso indica que a maioria dos usuários não têm in-

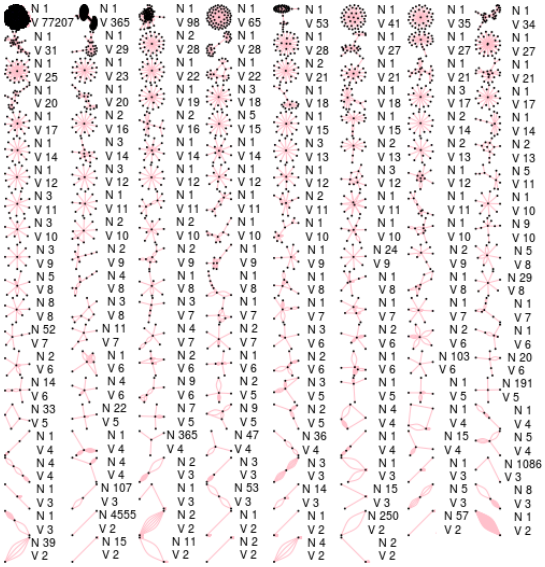


Figura 10. Estruturas de conexão entre usuários do OR que se retuítam

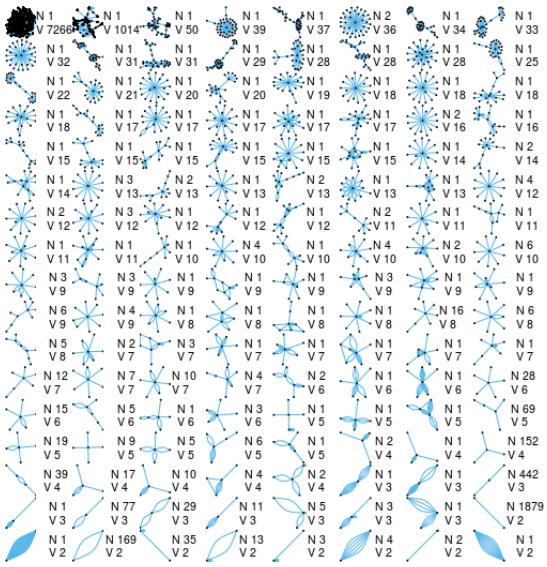


Figura 11. Estruturas de conexão entre usuários do NA que se retuítam

fluência para propagar os *tweets* na rede do Twitter. Ainda, em todos os pontos desta série, observa-se que o OR apresentou uma maior capacidade de propagação de informação comparado ao NA. Podemos identificar também que nas três (3) categorias de usuários, o OR prevalece sobre o NA em números de *retweets* para cada *tweet*. Para todas as categorias as diferenças ocorrem nos dois eixos, onde a frequência dos graus é maior para o OR, e também há casos no OR com *tweets* com grau não alcançado pelo NA. Assim, pode-se concluir que a abrangência de propagação dos *tweets* originais através de *retweets* é maior para o OR, pelo fato dos graus de *retweets* do OR serem mais frequentes do que o NA, e no OR os *tweets* atingirem graus de *retweets* que o NA não atinge.

Conclui-se assim que as campanhas não apresentam abrangência similar na rede. Além de mais numerosos, os usuários do OR possuem maior média de *tweets*. A propagação destes *tweets* também é maior na campanha do OR, pelo fato de o engajamento através de *retweets* ser maior no OR do que no NA tanto em quantidade de *retweets*, frequência, e também no tamanho das conexões de usuários através de *retweets* entre si.

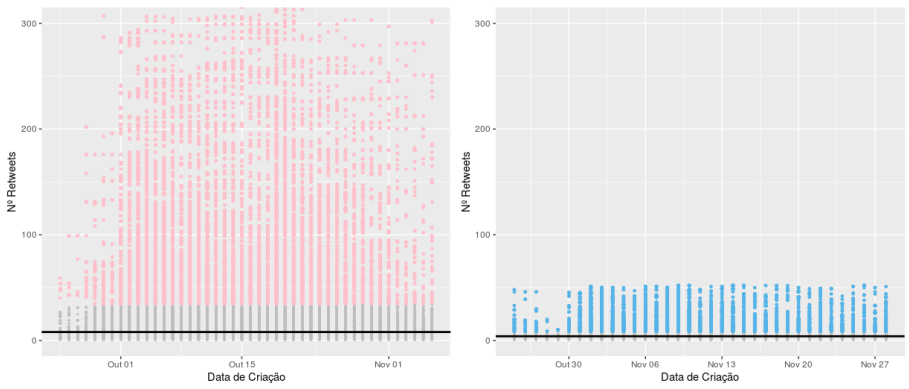


Figura 12. Frequência de Nº de Retweets por Data de Criação

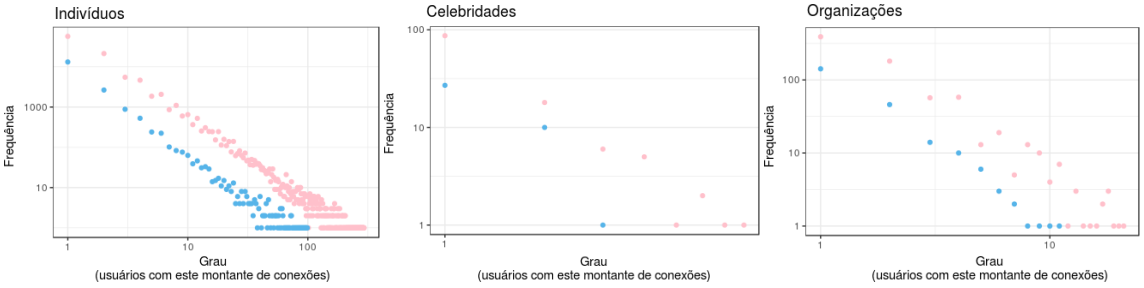


Figura 13. Distribuição de grau das conexões entre os usuários (log-log)

Estas conclusões se aplicam também em cada uma das categorias dos usuários, indicando que a influência de todas as categorias dos usuários é mais forte no OR.

5. Conclusão e Trabalhos Futuros

Providenciamos neste artigo algumas das primeiras intuições sobre uma comparação das campanhas do Outubro Rosa e Novembro Azul no contexto *online* no Twitter. Nas nossas análises, o Twitter providenciou boas reflexões de múltiplos aspectos das campanhas. Enquanto nossas análises mostram mais engajamento do público masculino, e que a faixa etária alvo das campanhas (homens e mulheres acima dos 40 anos) é a que mais participa no engajamento, identificamos que há diferenças de aplicação das campanhas nos diferentes países, sendo que os EUA e México não possuem um grande apelo para a campanha do Novembro Azul. Nós também percebemos que as estruturas de rede e engajamento da campanha do Outubro Rosa são maiores que a do Novembro Azul, possibilitando maior propagação da campanha na rede. Esperamos que este trabalho irá direcionar para um progresso nas campanhas do Novembro Azul, e que as mesmas possam alcançar números expressivos como as campanhas do Outubro Rosa. No futuro, pretendemos expandir nossas análises para avaliar os tópicos dos *tweets*, e realizar um estudo compartivo longitudinal.

References

Altekruse, S., Kosary, C., Krapcho, M., Neyman, N., Aminou, R., Waldron, W., Ruhl, J., Howlader, N., Tatalovich, Z., Cho, H., et al. (2010). Seer cancer statistics review, 1975–2007. *Bethesda, MD: National Cancer Institute*, 7.

Borgmann, H., Loeb, S., Salem, J., Thomas, C., Haferkamp, A., Murphy, D. G., and Tsaur, I. (2016). Activity, content, contributors, and influencers of the twitter discussion on urologic oncology. In *Urologic Oncology: Seminars and Original Investigations*, volume 34, pages 377–383. Elsevier.

Bravo, C. A. and Hoffman-Goetz, L. (2016). Tweeting about prostate and testicular cancers: Do twitter conversations and the 2013 movember canada campaign objectives align? *Journal of Cancer Education*, 31(2):236–243.

Castleton, K., Fong, T., Wang-Gillam, A., Waqar, M. A., Jeffe, D. B., Kehlenbrink, L., Gao, F., and Govindan, R. (2011). A survey of internet utilization among patients with cancer. *Supportive Care in Cancer*, 19(8):1183–1190.

Edge, L. (2006). Breast-cancer awareness: too much of a good thing? *Lancet Oncol*, 7:611.

ElSherief, M., Belding, E. M., and Nguyen, D. (2017). # notokay: Understanding gender-based violence in social media. In *ICWSM*, pages 52–61.

Fan, H., Cao, Z., Jiang, Y., Yin, Q., and Doudou, C. (2014). Learning deep face representation. *arXiv preprint arXiv:1403.2802*.

for Disease Control, C. and Prevention (2011). Cancer screening — united states, 2010. *Morbidity and Mortality Weekly Report (MMWR)*, pages 61(03); 41–45.

Glynn, R. W., Kelly, J. C., Coffey, N., Sweeney, K. J., and Kerin, M. J. (2011). The effect of breast cancer awareness month on internet search activity-a comparison with awareness campaigns for lung and prostate cancer. *BMC cancer*, 11(1):442.

Himelboim, I. and Han, J. Y. (2014). Cancer talk on twitter: community structure and information sources in breast and prostate cancer social networks. *Journal of health communication*, 19(2):210–225.

Jacobsen, G. D. and Jacobsen, K. H. (2011). Health awareness campaigns and diagnosis rates: evidence from national breast cancer awareness month. *Journal of health economics*, 30(1):55–61.

Jacobson, J. and Mascaro, C. (2016). Movember: Twitter conversations of a hairy social movement. *Social Media+ Society*, 2(2):2056305116637103.

Laranjo, L., Arguel, A., Neves, A. L., Gallagher, A. M., Kaplan, R., Mortimer, N., Mendes, G. A., and Lau, A. Y. (2014). The influence of social networking sites on health behavior change: a systematic review and meta-analysis. *Journal of the American Medical Informatics Association*, 22(1):243–256.

Lotan, G., Graeff, E., Ananny, M., Gaffney, D., Pearce, I., et al. (2011). The arab spring—the revolutions were tweeted: Information flows during the 2011 tunisian and egyptian revolutions. *International journal of communication*, 5:31.

Nastasi, A., Bryant, T., Canner, J. K., Dredze, M., Camp, M. S., and Nagarajan, N. (2017). Breast cancer screening and social media: a content analysis of evidence use and guideline opinions on twitter. *Journal of Cancer Education*, pages 1–8.

National Breast Cancer Foundation, I. (2017). Breast cancer awareness month.

Olteanu, A., Weber, I., and Gatica-Perez, D. (2016). Characterizing the demographics behind the# blacklivesmatter movement. *OSSM. http://arxiv. org/abs/1512.05671*.

Prasetyo, N. D., Hauff, C., Nguyen, D., van den Broek, T., and Hiemstra, D. (2015). On the impact of twitter-based health campaigns: A cross-country analysis of movember. In *Proceedings of the Sixth International Workshop on Health Text Mining and Information Analysis*, pages 55–63.

Schulz, A., Schmidt, B., and Strufe, T. (2015). Small-scale incident detection based on microposts. In *Proceedings of the 26th ACM Conference on Hypertext & Social Media*, pages 3–12. ACM.

Sloan, L., Morgan, J., Burnap, P., and Williams, M. (2015). Who tweets? deriving the demographic characteristics of age, occupation and social class from twitter user metadata. *PloS one*, 10(3):e0115545.

Thackeray, R., Burton, S. H., Giraud-Carrier, C., Rollins, S., and Draper, C. R. (2013). Using twitter for breast cancer prevention: an analysis of breast cancer awareness month. *BMC cancer*, 13(1):508.

Análise de colaboração em desenvolvimento global de software

Vitor A. C. Horta[1],Victor Ströele[1], Jonice Oliveira[2], Regina Braga[1], José Maria David[1], Fernanda Campos[1]

[1]Departamento de Ciência da Computação – Universidade Federal de Juiz de Fora (UFJF)
Caixa Postal 422, 36001-970 – Juiz de Fora – MG – Brazil

[2]COPPE/UFRJ - Computer Science Department - Graduate School and Research in Engineering – Federal University of Rio de Janeiro

Abstract. *The global open source software development popularity motivates the search for experts with capability of helping other developers in solving complex tasks. The challenge is: given a task, how to identify an expert (or set of experts) to execute it? This problem is named the expert-location problem. Some of these search difficulties are the large amount of data, the lack of technical details about the candidates and the different levels of collaboration. This work aims to detect experts and to identify groups with experienced members in some topics in Q&A forums. To achieve these goals the StackOverflow forum was used and modeled as a complex network. The presented method uses NetSCAN algorithm to detect overlapping communities in social networks. Through a temporal analysis the developers' skills were revealed. It was also found out that some users are changing their interests over time. The evaluation was conducted through a viability analysis by comparing the scores of the answers given by experts (indicated by the proposed method) and by common users.*

Resumo. *A popularidade do desenvolvimento global de software Open Source aumenta a necessidade de busca por especialistas capazes de auxiliar outros desenvolvedores na resolução de tarefas complexas. O desafio é: dada uma tarefa, como identificar o melhor especialista (ou um conjunto de especialistas) para executá-la? Este problema é chamado de localização de especialistas. Algumas das dificuldades desta pesquisa são o grande volume de dados produzido, a ausência de maiores detalhes técnicos dos envolvidos e os diferentes níveis de colaboração. O objetivo deste trabalho é detectar especialistas e identificar grupos com participantes experientes em determinados tópicos em um fórum Q&A. Para tal, o fórum StackOverflow foi modelado como uma rede complexa. O método apresentado é composto pela detecção de comunidades sobrepostas através do algoritmo NetSCAN. Através de uma análise temporal foram reveladas as aptidões dos desenvolvedores, mostrando também uma tendência de mudança de seus interesses. A avaliação do método foi feita através de uma análise de viabilidade, comparando as notas das respostas dos especialistas (apontados pelo método proposto) com as notas das respostas dos usuários comuns.*

1. Introdução

O aumento da demanda por software e o crescimento do desenvolvimento global de software despertam o interesse na busca por desenvolvedores experientes ou especialistas em determinados tópicos de desenvolvimento [Ma et al. 2009].

Esta busca por especialistas procura solucionar problemas como: roteamento de perguntas [Li and King 2010], correção de defeitos [Zhang and Lee 2012] e revisão de código fonte [Rahman et al. 2016]. O problema de roteamento de perguntas está relacionado à baixa taxa de respostas em fóruns Q&A e, uma forma de resolvê-lo, é encaminhar as perguntas para usuários que possuem maior chance de dar uma resposta relevante. A recomendação de desenvolvedores para correção de defeitos é outro problema que pode ser apoiado pela busca por especialistas. Segundo [Zhang and Lee 2012] o aumento do tamanho e da complexidade de software faz com que o número de *bugs* relatados em sistemas *open source* seja muito elevado. Uma forma de aumentar a taxa de correção destes defeitos é através da recomendação de desenvolvedores apropriados para a tarefa. A recomendação de desenvolvedores para processo de revisão de código também é outro problema encontrado em ambientes colaborativos. Essa atividade pode ser apoiada através da disponibilização de informações sobre as especialidades de desenvolvedores [Rahman et al. 2016].

Uma forma de identificar tais desenvolvedores é através da análise de fóruns Q&A (perguntas e respostas), tais como: Yahoo! answers, Quora e StackOverflow. Dentre eles, o StackOverflow se destaca por possuir uma grande quantidade de usuários, uma alta taxa de respostas e um pequeno tempo de resposta [Mamykina et al. 2011].

O StackOverflow possui uma grande quantidade de usuários ativos, mais de 5 milhões desde 2008 [Bayati 2016]. Promove a colaboração e troca de conhecimento entre essas pessoas através de perguntas e respostas no website. Algumas características importantes deste fórum são: a utilização de tags (palavras-chave) para determinar o domínio de cada conversação; e a pontuação de perguntas e respostas, que determinam a relevância de cada contribuição.

A identificação de desenvolvedores especialistas através de fóruns Q&A é um assunto que vem sendo abordado por diversos trabalhos [Li and King 2010] [Yang and Manandhar 2014] [Fu et al. 2017]. No entanto, muitos destes trabalhos propõem rankings de desenvolvedores por tópicos e não consideram elementos de colaboração em seus estudos.

Além disso, segundo Rubin [Rubin and Rinard 2016], fatores como diferença de fuso horário e de linguagem possuem grande impacto e podem reduzir a produtividade no trabalho colaborativo. Pessoas que já possuem colaborações em conjunto tendem a ser menos impactadas com estes problemas [Aggarwal 2011]. Desta forma, torna-se interessante identificar grupos de pessoas que obtiveram sucesso por colaborarem em alguma atividade. Entretanto, em função do grande volume de dados existentes nessas bases de fóruns Q&A, é problemático identificar e analisar as relações entre esses desenvolvedores, bem como identificar grupos de desenvolvedores que já tenham colaborado com sucesso em alguma atividade.

Neste sentido, este trabalho tem como objetivo identificar: (i) grupos de desenvolvedores que já tenham trabalhado colaborativamente em desenvolvimento de software, e (ii) desenvolvedores especialistas em determinados tópicos de desenvolvimento. Com isso, pretende-se identificar especialistas para resolução de problemas e pessoas capazes de trabalhar colaborativamente.

Para alcançar os objetivos deste trabalho, foi modelada uma rede de colaboração

entre desenvolvedores a partir das interações entre os usuários no StackOverflow. Após a construção dessa rede foi utilizado um método de detecção de comunidades e identificação de pessoas influentes em redes complexas, denominado NetSCAN [Vitor Horta 2017]. Com base nos resultados obtidos pelo algoritmo, foi feita uma análise detalhada da rede com o intuito de caracterizar a colaboração nos grupos identificados, e as especialidades dos desenvolvedores.

Este trabalho está organizado da seguinte forma: a seção 2 apresenta métodos para detecção de comunidades e desenvolvedores especialistas, a seção 3 detalha a modelagem da rede StackOverflow. A seção 4 discute o uso do algoritmo NetSCAN. As seções 5 e 6 mostram, respectivamente, os resultados obtidos e a avaliação dos mesmos, e na seção 7 são apresentadas as considerações finais e os trabalhos futuros.

2. Grupos de Colaboração no StackOverflow

No StackOverflow os usuários podem criar perguntas e atribuir até 5 tags a cada pergunta. Essas tags ajudam a definir o domínio da pergunta e o interesse dos usuários. Alguns exemplos de tags existentes são *"c" "java"*, *"javascript"*, *"sql"* e *"html"*. Para manter a consistência das tags o StackOverflow utiliza um sistema de recomendação de tags já existentes, além de permitir que apenas usuários mais experientes possam criar novas tags. Após a criação da pergunta os outros usuários do fórum podem então submeter uma resposta. Os próprios usuários do fórum podem votar positivamente ou negativamente nas perguntas e respostas indicando quais são as mais relevantes e com maior contribuição.

Uma característica deste tipo de fórum é que qualquer usuário pode responder e votar nas perguntas e respostas sem que haja um relacionamento pré-estabelecido entre eles. De acordo com [Meng et al. 2015] isto faz com que a estrutura de fóruns Q&A seja mais parecida com estruturas de estrela ao invés de estrutura de triângulos. Dessa forma, as pessoas formam grupos por tópicos de interesse e, por possuírem múltiplos interesses, estas pessoas participam de diferentes grupos com interesses distintos.

A identificação destes grupos pode ser alcançada através de métodos de detecção de comunidades. Para isso existem 3 principais tipos de abordagem [Meng et al. 2014]: métodos baseados em grafos, métodos de agrupamento e métodos baseados em modelos. A primeira consiste em inferir um grafo através das interações no fórum e, posteriormente, utiliza-se algum método de detecção de comunidades em redes [Xie et al. 2013][Cuijuan Wang and Wang 2015] no grafo inferido. Esta abordagem possui algumas limitações. Primeiramente, ela não utiliza os atributos dos nós e dos relacionamentos. Também não é possível saber os tópicos em que as pessoas interagiram e, portanto, não é possível identificar os tópicos de interesse dos usuários.

Outra forma de abordar este problema é através da utilização de métodos de agrupamento. Neste caso, calcula-se a similaridade dos perfis dos usuários e utiliza-se métodos de agrupamento baseados em similaridade [Meng et al. 2014]. Nesta abordagem a estrutura da rede não é considerada e cada usuário é atribuído a apenas um grupo. Esta é uma grande limitação no contexto de fóruns Q&A, já que os usuários, em geral, participam de múltiplos grupos com diferentes interesses.

Uma terceira forma de identificar grupos neste contexto é através de um modelo que considere tanto a estrutura da rede quanto os atributos dos nós e dos relacionamentos.

Esta abordagem é utilizada por [Meng et al. 2014] e [Kianian et al. 2017] e permite identificar os tópicos de interesse dos usuários, associar usuários a múltiplos grupos e detectar comunidades sobrepostas.

O modelo proposto neste trabalho se assemelha com esta terceira abordagem, mas, diferente dos modelos anteriores [Meng et al. 2014][Kianian et al. 2017], este considera que as relações entre os usuários são direcionadas e indicam a relevância das contribuições de um usuário para o outro. Assim é possível identificar quais são os usuários que fizeram as contribuições mais relevantes na rede permitindo a detecção de usuários especialistas e influentes em grupos de interesse. Outra característica deste trabalho é permitir que existam múltiplos grupos referentes a um mesmo tópico de interesse. Dessa forma, os grupos detectados além de indicarem usuários que compartilham o mesmo interesse também mostram usuários com maior potencial para futuras contribuições. Para tal foi utilizado o NetSCAN, um algoritmo para detecção de comunidades e usuários influentes em redes de grande porte.

3. Rede de Colaboração entre Desenvolvedores

A rede de desenvolvedores proposta neste trabalho foi modelada a partir das perguntas e respostas do StackOverflow. Os vértices da rede representam os usuários do fórum. As arestas representam todas as respostas dadas pelo vértice de origem para o vértice de destino em uma determinada tag.

Cada usuário pode responder várias vezes a uma mesma pergunta e, além disso, cada uma das respostas pode conter até 5 *tags*. Essas respostas recebem, cada uma delas, um *score* atribuído pelos usuários do StackOverflow.

Dessa forma, a rede de colaboração foi representada por um grafo direcionado $G = (V, E)$, onde $V = \{v_0, v_1, ..., v_{n-1}\}$ é o conjunto de *n* vértices (usuários), E representa o conjunto de arestas na forma $e_{ijt} = (v_i, v_j, t)$ entre os usuários v_i e v_j em uma determinada tag t. Como um usuário pode dar várias respostas contendo uma mesma *tag*, $R_{ijt} = \{r^0_{ijt}, r^1_{ijt}, ..., r^{l-1}_{ijt}\}$ é um conjunto de *l* respostas sobre a *tag* t entre os usuários v_i e v_j, sendo que r^k, para $0 \leq k < l$, possui o *score* dado pelos usuários à essa resposta.

Com o intuito de mensurar as contribuições entre usuários em cada *tag*, as arestas possuem um peso que representa a relevância de todas as contribuições que o usuário v_i realizou sobre v_j em uma *tag* t, definido por $IP(e_{ijt})$.

Para calcular o peso da aresta $IP(e_{ijt})$ obtém-se primeiramente o somatório dos *scores* (pontuação) de todas as respostas do usuário v_i para v_j na tag t. A Equação 1 define o cálculo deste somatório, onde $S(r^k_{ijt})$ é o *score* obtido pela k-*ésima* resposta dada por v_i a v_j em uma tag t e l é a quantidade de respostas dadas de v_i para v_j sobre a tag t.

$$Sum_{ijt} = \sum_{k=0}^{l-1} S(r^k_{ijt}) \tag{1}$$

A Equação 1 é normalizada dividindo o somatório dos *scores* Sum_{ijt} pelo total de contribuições recebidas por v_j na tag t. A Equação 2 define o cálculo do peso da aresta $IP(e_{i,j,t})$ onde $\parallel N(j,t) \parallel$ é a soma de todos *scores* recebidos por v_j na tag t.

$$IP(e_{ijt}) = \frac{Sum_{ijt}}{\parallel N(j,t) \parallel} \quad (2)$$

Como $Sum_{ijt} \leq \parallel N(j,t) \parallel$ então $0 \leq IP(e_{ijt}) \leq 1$. Desta forma, quanto mais $IP(e_{ijt})$ se aproxima de 1 maior a relevância das contribuições de v_i para v_j na tag t. Por outro lado, quanto mais $IP(e_{ijt})$ se aproxima de 0, menor será a relevância destas contribuições. Se $IP(e_{ijt})$ = 1 então v_i foi o único usuário a contribuir positivamente com v_j nesta tag. O valor de $IP(e_{ijt})$ também pode ser 0 caso v_i tenha contribuído com v_j mas a soma dos *scores* de suas contribuições são menores ou iguais a 0.

A Figura 1 mostra uma abstração desta rede onde o usuário v_i contribuiu com o usuário v_j em m *tags* distintas. É possível que v_i tenha dado múltiplas respostas para cada *tag*, já que cada aresta representa todas as contribuições em uma mesma *tag*. Apesar de não aparecerem na figura, a quantidade de respostas e o *score* total obtido por elas estão armazenados e podem ser acessados através dos atributos de cada aresta.

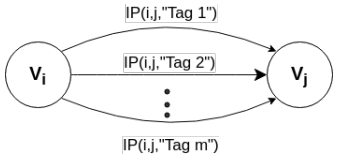


Figura 1. Interações entre dois desenvolvedores

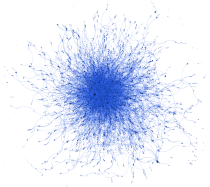


Figura 2. Visualização da rede oferecida pelo gephi

Para implementar a rede foi utilizado um *dataset* contendo dados do ano de 2008 até 2016 com 565680 usuários, 618726 perguntas (*posts*) e 1188765 respostas sobre as 20 *tags* mais frequentes do StackOverflow. A rede de desenvolvedores foi implementada através de um banco de dados orientado a grafos Neo4j (https://neo4j.com/). A Figura 2 mostra uma visualização total da rede feita com o *Gephi* (https://gephi.org/).

Como observado na Figura 2 não é viável fazer uma análise visual desta rede devido ao grande volume de dados. Neste sentido, o algoritmo NetSCAN foi utilizado para a detecção automática de grupos colaborativos e desenvolvedores especialistas.

4. Uso do NetSCAN para detecção de comunidades

Após modelar a rede e implementá-la no Neo4j foi definida uma estratégia para particionar o grafo que representa a rede e executar o algoritmo NetSCAN [Vitor Horta 2017] para detecção de comunidades em cada uma dessas partições.

O NetSCAN é um algoritmo de detecção de comunidades baseado em densidade que identifica vértices influentes (*cores*) na rede e detecta comunidades sobrepostas a partir destes *cores*. Além disso, o NetSCAN determina o número de grupos automaticamente e não limita o número de grupos que um vértice pode participar. Estas características são importantes no contexto de fóruns Q&A já que os usuários tendem a participar de vários grupos e que existem usuários especialistas em determinados tópicos de interesse.

A Figura 3 mostra um diagrama que ilustra o processo de particionamento e execução do algoritmo. O processo se inicia no passo 1 onde ocorre a implementação

da rede no Neo4j. No passo 2 uma *tag* t é selecionada e inicia-se um processo iterativo para o particionamento e execução do NetSCAN. No passo 3 é extraído o subgrafo $G_t \subset G$ que contém apenas os vértices e arestas relacionados a *tag* t. O subgrafo G_t é utilizado como entrada para a execução do NetSCAN no passo 4, onde são detectadas as comunidades e os usuários especialistas relacionados a *tag* t. O passo 5 é utilizado para decidir se novas iterações são necessárias e, caso não existam outras *tags*, o processo se encerra no passo 6.

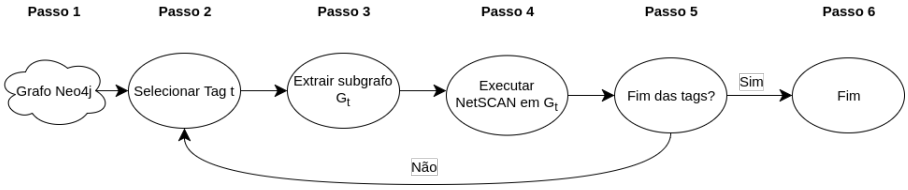


Figura 3. Diagrama para execução do algoritmo de detecção de comunidades

Ao final do processo definido na Figura 3 as comunidades detectadas são armazenadas no Neo4j por vértices do tipo *Cluster* e os usuários especialistas são os vértices do tipo *User* que possuem o atributo *core*. A próxima seção apresenta e analisa os resultados obtidos através deste processo.

5. Resultados

Nesta seção é apresentada uma análise dos resultados do processo de detecção de comunidades na rede do StackOveflow. Primeiramente foi feita uma análise exploratória dos resultados com o objetivo de levantar as principais características das comunidades e dos desenvolvedores especialistas detectados. Depois foi feita uma análise detalhada para entender o motivo real destas características.

5.1. Análise das Comunidades e desenvolvedores especialistas

Os resultados do processo de detecção de comunidades foram analisados com apoio de ferramentas de visualização e de consultas no Neo4j. As Figuras 4 e 5 mostram como são representadas as comunidades e seus membros no banco de dados.

A Figura 4 mostra uma comunidade de *python* (retângulo vermelho) com apenas um desenvolvedor especialista (vértice verde com seta), representado pelo vértice *core* central da figura. Os outros vértices azuis representam desenvolvedores que possuem interesse e participam de *posts* sobre *python*, mas não são considerados especialistas neste assunto. Assim como esperado essa comunidade possui estrutura de estrela na qual todas as ligações estão centralizadas no vértice *core*. A Figura 5 mostra uma comunidade de *c++* que contém dois desenvolvedores especialistas e que também possui estrutura de estrela. Neste caso os dois vértices *cores* desta comunidade possuem contribuições relevantes com outros usuários e também entre si.

Através das execuções do NetSCAN foram detectadas ao todo 10897 comunidades e 11996 desenvolvedores especialistas sendo que *"javascript"* foi a *tag* com maior número de comunidades (1043) e *"java"* a que possui o maior número de especialistas (1551).

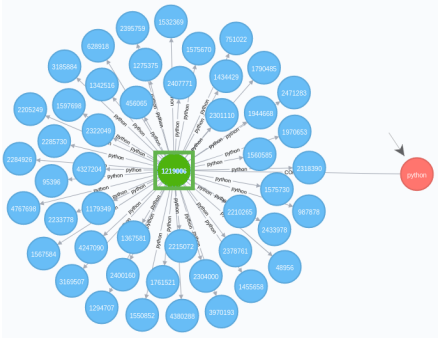


Figura 4. Comunidade (vértice vermelho com seta) de python com apenas um *core* (retângulo azul centralizado)

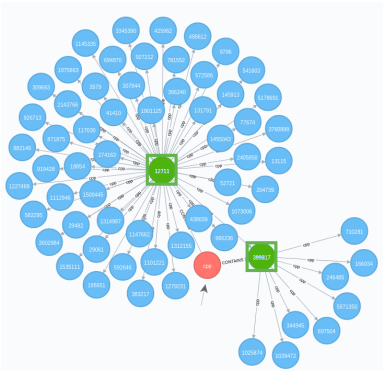


Figura 5. Comunidade de c++ com dois *cores*

Nota-se que o número total de especialistas é maior que o número total de comunidades. Isso acontece porque as comunidades podem possuir múltiplos especialistas que colaboraram entre si. Além disso, existem casos onde os especialistas participam de mais de uma comunidade, gerando sobreposições entre essas comunidades. A Figura 6 ilustra três casos de comunidades (vértices vermelhos) de diferentes tópicos de interesse que se sobrepõem através de um desenvolvedor especialista (vértice azul).

Figura 6. Sobreposição em comunidades de diferentes interesses

Estas sobreposições indicam que o desenvolvedor compartilhado entre as duas comunidades possui interesse e habilidades em ambas tecnologias, já que, para ser considerado um vértice *core* em múltiplas comunidades este usuário deve possuir colaborações relevantes nestes múltiplos assuntos.

Outra informação relevante que pode ser encontrada nas sobreposições entre as comunidades é a correlação entre dois assuntos, pois comunidades de assuntos distintos que compartilham múltiplos membros podem indicar uma alta correlação entre estes assuntos. A Tabela 1 mostra o número de usuários compartilhados entre comunidades de diferentes *tags*. Percebe-se através dessa tabela que os pares de *tags* com maior número de usuários compartilhados são de fato tecnologias muito relacionadas, sendo elas: *javascript* e jquery; *html* e *css* e; *android* e *java*. Pode-se dizer então que há indícios que de fato a quantidade de usuários compartilhados entre as comunidades indicam a correlação entre os interesses destas comunidades.

Tabela 1. Usuários compartilhados por comunidades de diferentes tags

Tag 1	Tag 2	Usuários compartilhados
”javascript”	”jquery”	9979
”html”	”javascript”	4804
”css”	”html”	4389
”android”	”java”	2476
...	...	...
”python”	”rubyonrails”	1
”iphone”	”mysql”	2

5.2. Análise temporal das sobreposições

Após perceber a existência de desenvolvedores especialistas que participam simultaneamente de múltiplas comunidades com diferentes tópicos de interesse, foi feita uma análise temporal sobre estes usuários para entender o motivo real destas sobreposições.

Nesta análise foram coletados os anos das respostas dadas pelos usuários sobrepostos em cada uma de suas comunidades. Foram detectados três principais casos que podem incentivar o surgimento das sobreposições: (i) desenvolvedor ativo em comunidades de assuntos muito relacionados; (ii) desenvolvedor ativo em comunidades de assuntos concorrentes e; (iii) desenvolvedor em processo de mudança de tópico de interesse.

O primeiro motivo é o mais comum e recorrente. Como as comunidades são caracterizadas por terem interesses em tecnologias muito relacionadas, os desenvolvedores especialistas em ambas tecnologias conseguem se manter ativos nas múltiplas comunidades. Para ilustrar este caso foi escolhido um usuário que participa ativamente de uma comunidade de *javascript* e uma comunidade de *jquery*. Percebe-se através da Figura 7 que o usuário analisado manteve sua atividade nas duas comunidades durante todo seu período de contribuição no fórum.

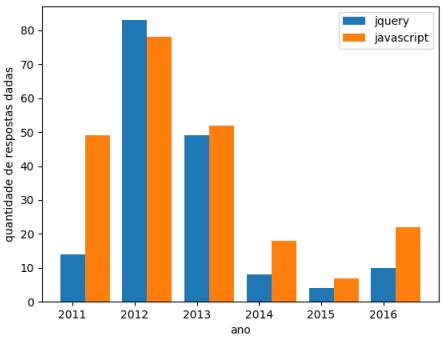


Figura 7. Histórico de atividade de um desenvolvedor especialista em comunidades de jquery e *javascript*

Outro motivo real de sobreposição encontrado acontece quando um usuário consegue se manter ativo em duas comunidades que possuem interesse em tecnologias concorrentes. Este caso é mais raro pois exige que o desenvolvedor sobreposto tenha habilidade em tecnologias pouco relacionadas ou assuntos muito distintos. A Figura 8 mostra o histórico de atividades de um usuário que se manteve ativo em uma comunidade de *android* e uma comunidade de *iOS*, mostrando assim a capacidade deste usuário em desenvolver em ambientes multiplataforma e utilizando diferentes linguagens de programação. Isto pode impactar positivamente na recomendação deste desenvolvedor, já que o mesmo pode exercer diferentes funções em um mesmo projeto.

Também é possível encontrar casos de sobreposições ocasionadas por usuários que mudaram de tópicos de interesse. Este cenário indica que o usuário envolvido já contribuiu com uma determinada comunidade no passado mas está colaborando em outros tópicos e com outras comunidades no momento atual. Este caso é mais frequente em comunidades de interesses distintos ou concorrentes, já que o desenvolvedor tende a ter mais

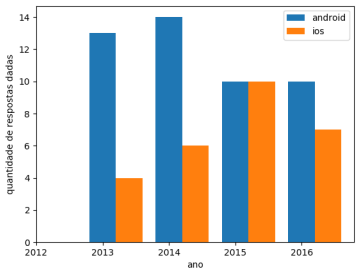


Figura 8. Atividade de um desenvolvedor especialista em comunidades de android e *iOS*

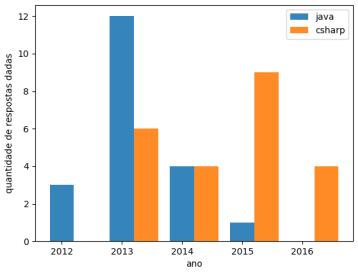


Figura 9. Transição de interesses de um desenvolvedor de java para *csharp*

dificuldade em se manter atualizado e ativo nas diferentes tecnologias e acaba migrando de comunidade.

No gráfico da Figura 9 é possível ver que o usuário esteve ativo em uma comunidade com interesse em *java* no início de suas colaborações, mas ao longo do tempo passou a contribuir com uma outra comunidade sobre *csharp*. Isto mostra que mesmo possuindo habilidades em *java* este desenvolvedor está mais apto no momento a participar de atividades e responder questões sobre *csharp*.

Foram encontrados também especialistas que participam de mais de duas comunidades, chegando a um máximo de 12 comunidades para um mesmo usuário. Estes casos porém são menos recorrentes e, dessa forma, não foi possível identificar suas principais motivações.

Os resultados desta análise temporal das sobreposições mostram que o estudo do histórico do usuário pode indicar a multidisciplinaridade deste desenvolvedor e seus interesses e aptidões atuais, podendo auxiliar em processos de tomada de decisão como a recomendação de desenvolvedores e roteamento de perguntas.

6. Avaliação

Nesta seção foi feita uma análise com dados recentes do StackOverflow para descobrir se os especialistas encontrados pelo NetSCAN são de fato pessoas ativas e com alto conhecimento nos tópicos de interesse relacionados. Para isso foi selecionado um conjunto de teste contendo as respostas dadas por usuários especialistas e não especialistas (apontados pelo método proposto) nos anos 2017 e 2018. Dessa forma nenhuma resposta deste conjunto de teste foi utilizada previamente no agrupamento realizado pelo NetSCAN.

Os usuários foram separados em dois grupos A (especialistas) e B (não especialistas), cada grupo contendo 20 usuários escolhidos aleatoriamente. Para cada usuário foi calculada a média dos scores de suas respostas mais recentes. As médias obtidas por cada grupo foram então comparadas com o intuito de descobrir se o desempenho alcançado pelos especialistas foi maior que dos não especialistas.

O *boxplot* da Figura 10 mostra o desempenho alcançado por usuários dos grupos A e B deste conjunto de teste. Pode-se perceber através do gráfico que os *scores* das respostas dadas pelos especialistas são superiores as respostas dos não especialistas. O *outlier* pertencente ao grupo A refere-se a um usuário que além de ter muitas respostas

recentes possui uma resposta com score de 262, sendo esta muito acima da média. Por outro lado os dois outliers do grupo B são usuários que tiveram uma resposta com *score* acima da média mas a maioria de suas outras respostas possuem *score* igual a 0. Além disso ambos os grupos possuem usuários com média igual a 0, que são usuários que não deram nenhuma resposta no período analisado ou não obtiveram nenhum *score* positivo em suas respostas.

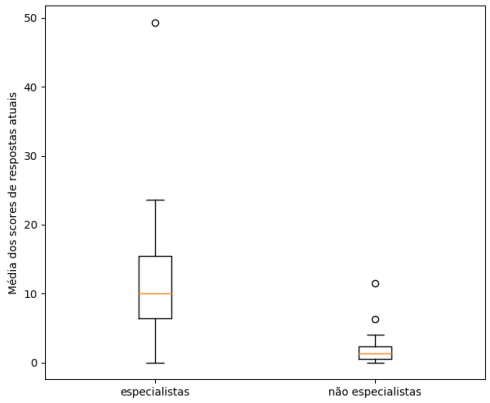


Figura 10. Desempenho dos usuários especialistas e não especialistas

Para avaliar se existe uma diferença significativa na média do desempenho dos dois grupos, os *scores* dos grupos A e B foram submetidos a um teste estatístico. Primeiramente verificou-se a não normalidade dos dados através do teste de *Kormogorov-Smirnov*. Depois foram levantadas duas hipóteses H0 (hipótese nula) e H1 (hipótese alternativa):

H0: As médias dos *scores* dos grupos A e B são iguais.

H1: As médias dos *scores* dos grupos A e B são diferentes.

Os *scores* foram então comparados pelo teste de *Mann-Whitney* com nível de confiança de 95%. Como resultado foi encontrado *p-value* $< 0,05$ e, dessa forma, rejeitou-se a hipótese nula de que as médias são iguais e aceitou-se a hipótese alternativa de que as médias são diferentes. Como a média do grupo A é maior que a do grupo B, aceitou-se que a média dos especialistas é maior que a média dos não especialistas.

A partir da análise deste conjunto de teste é possível perceber que as respostas dadas pelos usuários detectados como especialistas pelo NetSCAN possuem uma maior aceitação do que os usuários não especialistas. Este resultado aponta que o NetSCAN foi capaz de detectar tais usuários com maior *expertise* em determinados tópicos de interesse.

7. Considerações finais

Neste trabalho foi feita uma análise de redes sociais sobre o fórum Q&A de desenvolvimento de software StackOverflow com o objetivo de encontrar desenvolvedores especialistas e grupos colaborativos com desenvolvedores experientes. Para isso foi modelada uma rede social utilizando as respostas dadas pelos usuários do fórum. As *tags* de cada resposta foram utilizadas para definir o contexto de cada conversação e o *score* foi usado para medir a relevância das contribuições entre os usuários.

Para identificar grupos de desenvolvedores e usuários especialistas em cada tópico de interesse foi utilizado o algoritmo NetSCAN para detecção de comunidades e usuários influentes em redes sociais. Em um processo iterativo o grafo que representa a rede foi particionado em subgrafos relacionados a cada *tag* existente na rede e então o NetSCAN foi executado em cada um destes subgrafos.

Como resultados deste processo foram identificados grupos de desenvolvedores que colaboraram entre si em determinados tópicos de interesse e usuários especialistas nestes tópicos. Foram encontradas comunidades com múltiplos participantes influentes e comunidades sobrepostas com diferentes interesses. Em uma análise temporal identificou-se a existência de desenvolvedores multidisciplinares que atuam simultaneamente com tecnologias semelhantes e outros que atuam com tecnologias concorrentes. Também foram encontrados casos de usuários que começaram sua atividade no fórum colaborando sobre um tópico de interesse e depois migraram para outro tópico de interesse concorrente. Com base nestes resultados, os desenvolvedores especialistas podem ser recomendados para realização ou colaboração em tarefas complexas em desenvolvimento global de software.

A avaliação da proposta deste trabalho foi feita através da comparação do desempenho dos usuários especialistas detectados pelo NetSCAN com o dos usuários não especialistas. Para isso foi coletado um conjunto de teste contendo respostas mais recentes do StackOverflow e os *scores* destas respostas foram utilizados para medir o desempenho dos usuários. Através da análise de *boxplot* e de testes estatísticos pôde-se perceber que as respostas dadas pelos especialistas possuem maior aceitação do que as respostas dos não especialistas, o que aponta para a viabilidade da solução.

Como o um conjunto de teste coletado não contemplou toda a rede utilizada, outras formas de avaliação se fazem necessárias. Assim, como trabalhos futuros pretende-se avaliar os resultados junto à comunidade através de formulários e entrevistas e utilizar um conjunto de teste maior para a avaliação.

Referências

Aggarwal, C. C., editor (2011). *Social Network Data Analytics*. Springer US.

Bayati, S. (2016). Security expert recommender in software engineering. In *Proceedings of the 38th International Conference on Software Engineering Companion*, ICSE '16, pages 719–721, New York, NY, USA. ACM.

Cuijuan Wang, Wenzhong Tang, B. S. J. F. and Wang, Y. (2015). Review on community detection algorithms in social networks. In *2015 IEEE International Conference on Progress in Informatics and Computing (PIC)*, pages 551–555.

Fu, C., Zhou, M., Xuan, Q., and xiang Hu, H. (2017). Expert recommendation in oss projects based on knowledge embedding. *2017 International Workshop on Complex Systems and Networks (IWCSN)*, pages 149–155.

Kianian, S., Khayyambashi, M. R., and Movahhedinia, N. (2017). Fuseo: Fuzzy semantic overlapping community detection. *Journal of Intelligent Fuzzy Systems*, 32(6):3987–3998.

Li, B. and King, I. (2010). Routing questions to appropriate answerers in community question answering services. In *Proceedings of the 19th ACM International Con-*

ference on Information and Knowledge Management, CIKM '10, pages 1585–1588, New York, NY, USA. ACM.

Ma, D., Schuler, D., Zimmermann, T., and Sillito, J. (2009). Expert recommendation with usage expertise. In *2009 IEEE International Conference on Software Maintenance*, pages 535–538.

Mamykina, L., Manoim, B., Mittal, M., Hripcsak, G., and Hartmann, B. (2011). Design lessons from the fastest q&a site in the west. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 2857–2866, New York, NY, USA. ACM.

Meng, Z., Gandon, F., Faron Zucker, C., and Song, G. (2014). Empirical Study on Overlapping Community Detection in Question and Answer Sites. In *Advances in Social Networks Analysis and Mining (ASONAM), 2014 IEEE/ACM International Conference on*, Beijing, China.

Meng, Z., Gandon, F., and Zucker, C. F. (2015). Simplified detection and labeling of overlapping communities of interest in question-and-answer sites. In *2015 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, volume 1, pages 107–114.

Rahman, M. M., Roy, C. K., and Collins, J. A. (2016). Correct: Code reviewer recommendation in github based on cross-project and technology experience. In *2016 IEEE/ACM 38th International Conference on Software Engineering Companion (ICSE-C)*, pages 222–231.

Rubin, J. and Rinard, M. (2016). The challenges of staying together while moving fast: An exploratory study. In *Proceedings of the 38th International Conference on Software Engineering*, ICSE '16, pages 982–993, New York, NY, USA. ACM.

Vitor Horta, Victor Ströele, Fernanda Campos. José Maria N. David. Regina Braga. (2017). Redes sociais científicas: análise topológica da influência dos pesquisadores. *Sbbd proceedings 32nd Brazilian Symposium on Databases*.

Xie, J., Kelley, S., and Szymanski, B. K. (2013). Overlapping community detection in networks: The state-of-the-art and comparative study. *ACM Comput. Surv.*, 45(4):43:1–43:35.

Yang, B. and Manandhar, S. (2014). Exploring user expertise and descriptive ability in community question answering. In *2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014)*, pages 320–327.

Zhang, T. and Lee, B. (2012). How to recommend appropriate developers for bug fixing? In *2012 IEEE 36th Annual Computer Software and Applications Conference*, pages 170–175.

Caracterização topológica de redes viárias por meio da análise de vetores de características e técnicas de agrupamento

Gabriel Spadon, Lucas C. Scabora, Marcos R. Nesso-Jr, Caetano Traina-Jr, Jose F. Rodrigues-Jr

Instituto de Ciências Matemáticas e de Computação (ICMC)
Universidade de São Paulo (USP)
São Carlos, SP – Brazil

{spadon, lucascsb, marcosnesso}@usp.br, {caetano, junio}@icmc.usp.br

Abstract. *Complex networks contribute to computational research by their ability to design systems modeled with vertices and edges. They provide means to describe urban structures through their street mesh, expressing predicates that refer to the flow and transportation in an urban zone. Towards the analysis of information from street networks, and by means of metrics from its elements, this paper aims at describing interactions between different cities using feature vectors. Our analysis is based on the use of digital maps; through them, we provide means for data modeling and feature extraction to support analytical activities. Our results are based on the analysis of 645 cities, which shape the Brazilian state of Sao Paulo. We show how the joint of features from complex-network metrics can describe urban indicators that are rooted in the network topology and how they can reveal differences among cities.*

Resumo. *As redes complexas contribuem para a pesquisa computacional por sua capacidade de projetar sistemas modelados por vértices e arestas. Eles fornecem meios para descrever estruturas urbanas por meio das malhas viárias, expressando predicados que se referem ao fluxo e ao transporte em zonas urbanas. Este trabalho tem o objetivo de descrever as interações entre diferentes cidades usando seus vetores de características pela análise de informações viárias e das métricas inerentes aos seus elementos. Propõe-se uma análise baseada no uso de mapas digitais, pois permitem abordagens para modelagem de dados e extração de características que suportam atividades analíticas. Os resultados deste trabalho são baseados na análise de 645 cidades, que formam o estado brasileiro de São Paulo; tais resultados demonstram como características extraídas por métricas de grafos descrevem indicadores urbanos que estão enraizados na topologia da rede, e como podem revelar diferenças entre cidades distintas.*

1. Introdução e Trabalhos Relacionados

As redes complexas são usadas para modelar sistemas reais e sintéticos, sendo exemplos disso as redes de interação proteica, as malhas viárias e as linhas metroviárias. Essas redes, como modelos matemáticos, se destacam devido às suas propriedades algébricas e potencial computacional, com aplicabilidade analítica para suportar processos cognitivos de tomada de decisão (Boccaletti et al. 2006). Por meio de métricas e métodos baseados

na topologia e/ou geometria das redes é possível identificar características de interesse que não são óbvias por inspeções humanas; porque as redes podem ser grandes (elevado número de vértices), intrincadas (elevado número de arestas), ou podem conter padrões e atributos não triviais, cuja observação depende da aplicação de técnicas algorítmicas.

No caso específico da representação de malhas viárias, as redes complexas descrevem fatores relacionados ao deslocamento de indivíduos, a localização e alocação de serviços, a melhoria de tarefas relacionadas ao transporte e até ao estudo de fatores advindos do comportamento coletivo. Neste contexto, foi observada a falta de estudos e/ou análises que caracterizam os grupos de cidades por meio da similaridade de suas características estritamente topológicas, que é o objetivo deste trabalho. Essa abordagem tem aplicações para a compreensão da morfologia urbana, bem como para identificar o porquê cidades partilham propriedades por estarem próximas ou distantes entre si.

A proposta deste trabalho se baseia na análise de 645 cidades do estado de São Paulo, visando fornecer compreensão sobre as peculiaridades existentes em diferentes cidades, interpretando suas características globais e usando métodos da área de aprendizado de máquina para a modelagem de dados, análise de agrupamentos e projeção multidimensional. Neste cenário, as seguintes premissas motivaram a presente pesquisa: **(A)** a topologia da rede é um poderoso conjunto de ferramentas que pode ser usado para identificar grupos de cidades com características semelhantes, potencialmente revelando disparidades (cidades que são muito grandes ou muito pequenas) sem utilizar dados demográficos; **(B)** embora cidades possam compartilhar fronteiras administrativas com outras, elas tendem a se agrupar com cidades das quais estão distantes; e, **(C)** pode haver correlação entre indicadores urbanos e/ou territoriais quando comparados com as características extraídas da topologia das redes viárias dentre o conjunto de 645 cidades.

Com o objetivo de resolver questões relacionadas ao cenário urbano, estudos foram realizados para descrever cidades considerando seu intenso fluxo de veículos (Masucci et al. 2013) e comportamento coletivo (Blumer 1971), enquanto outros analisaram a densidade de acidentes nas redes viárias (Anderson 2009) e as discrepâncias entre cidades por meio de seus indicadores urbanos (Grauwin et al. 2015). Alguns autores investigaram métodos métrico-analíticos aplicados em cidades (Crucitti et al. 2006, Costa et al. 2010), outros se concentraram no apoio ao desenho e ao planejamento urbano (Porta et al. 2009, Strano et al. 2012, Spadon et al. 2017), e há aqueles que avançaram com a análise e posicionamento de instalações (centros de serviço públicos e/ou privados) em cidades (Li and Parrott 2016). Entretanto, dentre estas aplicações, a análise de agrupamentos é ainda incipiente, mas é considerada um poderoso conjunto de ferramentas (Pan et al. 2013).

Com propósito semelhante ao deste trabalho, duas pesquisas do estado da arte (Strano et al. 2013, Domingues et al. 2017) usaram técnicas de agrupamento para analisar grupos de cidades. A primeira teve a intenção de medir graus de semelhança entre dez cidades europeias, enquanto a segunda realizou uma avaliação de agrupamentos considerando a proximidade e sobreposição de 1150 cidades, principalmente da América anglo-saxônica. Diversos algoritmos são capazes de agrupar dados, por exemplo, alguns fornecem melhores resultados para os dados que estão dispostos em polígonos convexos, outros não convexos; ainda, existem aqueles que se baseiam na hierarquia dos dados, enquanto outros não. Todavia, ambos os autores não discutiram a significância de seus

respectivos resultados, ou seja, a qualidade dos agrupamentos, comprovando a adequação dos métodos aos dados. Algumas das métricas com este objetivo são: *Silhouette*, *Dunn Index*, *Z-Score*, *Accuracy*, e *Precision-Recall* (Kremer et al. 2011); cada uma delas avalia diferentes perspectivas dos agrupamentos e sua combinação pode revelar melhores resultados ao descrever os dados.

Este trabalho contribui com técnicas que promovem a análise de sistemas urbanos por meio de grafos. Os resultados têm aplicações para a compreensão de semelhanças e diferenças entre cidades. Para apresentar as contribuições, este artigo está organizado como segue: a Seção 2 expõe a proposta e explica a validação dos resultados; a Seção 3 discute os resultados sobre a aplicabilidade dos métodos propostos; e, por fim, a Seção 4 apresenta as conclusões e considerações finais.

2. Proposta

Nesta seção é apresentada a proposta deste trabalho, cujo objetivo é promover a comparação entre cidades e o agrupamento delas. Essa seção está dividida do seguinte modo: na Seção 2.1 apresentam-se notações formais sobre as redes viárias e vetores de características; na Seção 2.2 detalha-se a modelagem e pré-processamento de dados; na Seção 2.3 discute-se a extração de características; e, por fim, na Seção 2.4 descrevem-se as técnicas utilizadas para a mineração e avaliação dos vetores, visando promover a projeção e o agrupamento dos vetores de características extraídos.

2.1. Conceitos Fundamentais

Grafos direcionados e ponderados são referidos neste texto como redes complexas e, apesar de diferentes, redes complexas e grafos são considerados equivalentes. Todo grafo $\mathbf{G} = \{V, E\}$ é composto de um conjunto de $|V|$ vértices (ou nós) e outro de $|E|$ arestas. Além disso, cada aresta $e \in E$ é conhecida por ser um par ordenado $\langle o, d \rangle$, em que $o \in V$ é o nó de *origem* e $d \in V$ é o de *destino*, $o \neq d$. Para cada aresta pode ser atribuído um peso numérico (d_{od}), o qual é referente à *distância dos grandes círculos* entre os vértices o e d na projeção esférica da superfície da Terra (Konstantopoulos 2012).

Um vetor de características $A = (a_1, \ldots, a_n) \in \mathbb{R}^n$ pode conter múltiplas métricas extraídas de uma rede complexa. A comparação entre dois vetores A e B é baseada em uma função de distância pré-definida $f(A, B) : A \times B \rightarrow [i, j],\ i \leq j$, no qual i indica os vetores mais próximos e j os mais distantes; por exemplo, a função de distância *Minkowski* é definida como $f(A, B) = \sqrt[p]{\sum_{i=1}^{n} |a_i - b_i|^p}$, onde diferentes valores de p indicam funções de distância distintas, por exemplo, a *Manhattan* (p_1) e a *Euclidiana* (p_2).

2.2. Aquisição e Preparação dos Dados

Para cada uma das 645 cidades do estado de São Paulo, obteve-se seus limites administrativos, indicadores territoriais e demográficos por meio do Instituto Brasileiro de Geografia e Estatística (IBGE)[1]. Os dados utilizados para modelar as redes complexas foram extraídos do OpenStreetMap (OSM)[2], uma rede social de mapeamento colaborativo de vias. Os limites territoriais foram utilizados para segmentar os dados geográficos do OSM em pequenas porções, cada qual representando uma cidade. Cada uma destas

[1] ⟨www.ibge.gov.br⟩
[2] ⟨www.openstreetmap.org⟩

porções descrevem o mundo real por meio de objetos georreferenciados. Estes objetos são descritos por relações, as quais se referem às vias e aos cruzamentos entre elas.

Existem duas possibilidades para construir uma rede complexa a partir de um arquivo do OSM. O primeiro é conhecido como *Grafo Primal* (Porta et al. 2006b), o qual considera as ruas como arestas e seus cruzamentos como vértices. O outro é o *Grafo Dual* (Porta et al. 2006a) em que as ruas são vértices e os cruzamentos são arestas. Levando em consideração que os atributos espaciais são essenciais para o domínio urbano e que não se pode calcular distância (em metros) por meio de dados não espaciais, foi escolhido o *Grafo Primal* ao invés do *Grafo Dual*. Consequentemente, usando um *Grafo Primal* assume-se que as redes são planares e que podem ser representadas em duas dimensões, no qual uma ou mais arestas se cruzam somente onde nós são definidos.

2.3. Extração e Seleção de Características

Métricas de grafos podem ser divididas entre locais e globais (Scripps et al. 2010); as métricas locais descrevem as propriedades para cada um dos elementos que formam a rede, enquanto as métricas globais caracterizam toda a rede por um valor que descreve todos seus elementos em conjunto. Note que este trabalho faz uso das métricas globais pois permitem a comparação direta de cidades distintas, enquanto as métricas locais não.

Para obter os resultados destas métricas, foi desenvolvido um extrator de características que produz um vetor de características para qualquer rede complexa. Em um primeiro momento, foram selecionadas várias métricas de grafos como potenciais candidatos para prover características das cidades. Destas, 29 foram mantidas por sua capacidade de prover informações sobre redes viárias. Todas as métricas selecionadas têm base na análise da topologia da rede, uma vez que a topologia descreve a malha viária das cidades, que é a base para o desenvolvimento desta pesquisa.

As métricas que compõem o conjunto de testes são: **(1)** número de auto conexões; **(2)** número de nós; **(3)** número de arestas; **(4)** número de vias unidirecionais; **(5)** número de vias bidirecionais; **(6)** média do grau de entrada; **(7)** média do grau de saída; **(8)** grau médio; **(9)** grau ponderado de entrada; **(10)** grau ponderado de saída; **(11)** entropia da distribuição de grau; **(12)** coeficiente de correlação do grau dos nós; **(13)** média ponderada de distâncias da rede; **(14)** média das distâncias das vias; **(15)** raio da rede; **(17)** diâmetro da rede; **(17)** entropia da distribuição de distâncias; **(18)** entropia da distribuição de caminhos mínimos; **(19)** média dos caminhos mínimos; **(20)** densidade; **(21)** densidade de rede planar; **(22)** transitividade; **(23)** índice de agrupamento médio; **(24)** índice de agrupamento global; **(25)** dominância do ponto central; e, coeficiente de assortatividade de **(26)** entrada$\times$entrada, **(27)** entrada$\times$saída, **(28)** saída$\times$entrada, e **(29)** saída$\times$saída.

Após coletar todas as métricas, as não relevantes foram removidas por meio da análise de correlação. Foi calculado o coeficiente de correlação de Pearson (Chiang 2003) para cada par de métricas. Esse coeficiente é definido no intervalo $[-1, 1]$, no qual os valores extremos indicam, respectivamente, a correlação máxima negativa e positiva, enquanto 0 indica nenhuma correlação linear. Na sequência, foram definidos dois valores-limite $\left[-\frac{1}{2}, \frac{1}{2}\right]$ dentro do intervalo original, permitindo remover todos as características com forte correlação mútua. Nos casos em que duas características estão fora do intervalo de corte, uma das métricas foi descartada aleatoriamente. O processo de seleção garante que apenas métricas não relacionadas umas com as outras serão usadas para

descrever as cidades. Como resultado, cada vetor de características é definido como $F = (\mathcal{H}, \mathscr{L}, \mathcal{R}, \mathcal{E}, \mathcal{D}, \mathcal{P}, \mathcal{B}, \mathcal{G}_c)$, contendo apenas 8 das 29 métricas avaliadas. As 8 métricas escolhidas são definidas de acordo com Costa et al. 2007 como se segue:

Entropia da Distribuição de Grau **($\mathcal{H}$).** A distribuição de grau de uma rede descreve seus vértices por meio de probabilidades de acordo com a quantidade dos vértices que possuem o mesmo grau. Considerando que, a entropia representa a quantidade de incerteza e aleatoriedade em uma determinada informação, ao usar a entropia em uma distribuição de grau de uma cidade, pode-se medir a incerteza entre as conexões de suas vias. A métrica é descrita na Equação 1, onde P_k é a proporção de vértices com grau k.

Média dos Caminhos Mínimos **($\mathscr{L}$).** Quantifica a média de todos os caminhos mínimos (d_{ij}^S) que conectam todos os pares de vértices em um grafo (ver Equação 2). É usada para quantificar a capacidade de locomoção por meio dos caminhos mais curtos de uma cidade.

Coeficiente de Assortatividade **($\mathcal{R}$).** Refere-se ao grau de correlação entre pares de nós. Valores positivos indicam que os nós com grau similar tendem a se conectar uns aos outros, enquanto que valores negativos indicam o mesmo, mas em relação a nós com graus diferentes. Pode ser entendida como a probabilidade de passar de uma rua sem importância para uma rua importante, com base apenas no número de ruas adjacentes a ambas. A métrica (ver Equação 3) usa e_{xy} para indicar à fração de arestas que conectam nós com grau x e y, a_x e b_y para a fração de arestas que começam e terminam em vértices com grau x e y; e σ_a e σ_b para o desvio padrão das distribuições de a_x e b_y.

$$\mathcal{H} = -\sum_{k=0}^{\infty} P_k \times \log(P_k) \quad (1) \qquad \mathscr{L} = \frac{\sum_{i=1}^{|V|}\sum_{j=1}^{|V|} d_{ij}^S}{|V|(|V|-1)} \quad (2) \qquad \mathcal{R} = \frac{\sum_{xy} xy(e_{xy} - a_x b_y)}{\sigma_a \sigma_b} \quad (3)$$

Excentricidade **($\mathcal{E}$).** Esta métrica é local e mede para um conjunto de vértices o maior caminho mínimo entre todos os outros vértices do grafo (ver Equação 4). Em uma perspectiva global, a maior excentricidade de um grafo é conhecida como o *diâmetro*, enquanto a menor é denominada de *raio*. O diâmetro e o raio podem indicar cidades que sofrem com problemas de locomoção urbana, este é o caso de redes geograficamente esparsas, que são redes viárias que possuem o raio muito pequeno em relação ao diâmetro.

Densidade da Rede Planar **($\mathcal{D}$).** A densidade (ver Equação 5) de um grafo planar é definida como a relação entre o número de arestas $|E|$ e o número de todas as arestas possíveis em um grafo com $|N|$ nós, no qual as arestas não se cruzam a não ser nos nós da rede; é capaz de revelar o quão densa é a malha viária de uma cidade ou de um bairro.

Dominância do Ponto Central **($\mathcal{P}$).** A métrica avalia a centralidade global de uma rede por meio do desvio padrão entre os valores de *Betweenness* de seus vértices, que é uma métrica de centralidade baseada em distância. Para valores próximos de 0, sabe-se que

existem muitas rotas eficientes que são semelhantes às mais curtas; enquanto que, para valores próximos de 1, a métrica indica que a rede é vulnerável sem o nó central pois o mesmo é usado para conectar diferentes componentes, servindo como ponto de acesso (*e.g.* pontes, viadutos e túneis). Na Equação 6, usa-se $\bar{v}$ como o vértice com o maior *Betweenness* e $\mathcal{B}(v)$ como o *Betweenness* normalizado do vértice v, definido entre $[0, 1]$.

$$\mathcal{E}_i = \frac{1}{max\{d_{ij}^S | \forall j \in V\}} \quad (4) \qquad \mathcal{D} = \frac{|E| - |N| + 1}{2|N| - 5} \quad (5) \qquad \mathcal{P} = \frac{\sum_v^{|V|} \mathcal{B}_{\bar{v}} - \mathcal{B}_v}{|V|(|V| - 1)} \quad (6)$$

***Vias Bidirecionais* ($\mathcal{B}$).** Consiste no número de arestas bidirecionais de um grafo; elas representam vias que fornecem rotas em dois sentidos entre o mesmo par de vértices.

***Agrupamento Global* ($\mathcal{G}_c$).** Esta métrica consiste na fração do número de triângulos $\mathbb{N}_\triangle$ e triplas $\mathbb{N}_3$, que é dado por $\mathcal{G}_c = (3 \times \mathbb{N}_\triangle) \div \mathbb{N}_3$. Descreve como as ruas tendem a se agrupar nos cruzamentos de uma determinada cidade, de modo que quanto maior o valor, maiores as possibilidades de locomoção em menos etapas entre pares de vértices distintos.

2.4. Análise de Vetores de Características

Esta etapa concentrou-se na aplicação de dois métodos da literatura de mineração de dados: o primeiro de projeção multidimensional e o segundo de análise de agrupamentos. A projeção multidimensional permite a visualização de dados, reduzindo seu espaço dimensional, revelando particularidades e comportamentos a serem explorados por meio de análise de suas relações. A análise de agrupamentos, por sua vez, se concentra no estudo das interações entre os dados, inferindo que dois elementos são semelhantes porque estão no mesmo grupo ou são dissimilares porque estão em grupos distintos. Deste modo, a combinação destes dois métodos contribui para a avaliação das cidades pelo seu elevado potencial de revelar características e padrões intrínsecos ao conjunto de dados.

Em relação à projeção multidimensional, foram aplicadas duas técnicas de redução de dimensionalidade (Spiwok et al. 2015); a primeira é chamada Isomap e a segunda é conhecida como Análise de Componentes Principais (PCA). Isomap é uma técnica de redução de dimensionalidade não linear, que fornece uma projeção em uma dimensão inferior, mantendo a distância geodésica entre os dados. PCA é uma técnica linear que usa conversões ortogonais para transformar um conjunto de variáveis em valores linearmente não correlacionados com a maior variância mútua. Na análise de agrupamentos foi usado KMeans (MacQueen et al. 1967), que divide os dados em grupos de igual variância, minimizando a distância da soma de quadrados entre eles.

Para escolher as duas técnicas de projeção, usou-se conhecimento sobre o domínio; manteve-se registro de algumas cidades discrepantes já conhecidas, buscando abordagens que as diferenciasse nitidamente. Já a técnica de agrupamento foi escolhida por ser um algoritmo amplamente utilizado na literatura relacionada que é conhecido por ser escalável para um grande número de amostras e que tem sido usado para uma variedade considerável de propósitos em muitos domínios de aplicação.

Para a validação dos resultados, consideram-se métricas de avaliação de qualidade de agrupamentos (Kremer et al. 2011). O foco de tais métricas é analisar a semelhança

entre elementos que foram atribuídos ao mesmo grupo. As que atendem a este objetivo são: *Silhouette* e *Dunn Index*; ambos conhecidos como métricas de qualidade interna, as quais não demandam dados rotulados. A medida de *Silhouette* é definida no intervalo $[-1, 1]$; os valores são atribuídos a cada grupo inerente aos dados e quanto mais próximo de 1, melhor. *Dunn Index* foi usado para evitar casos em que a medida de *Silhouette* falha, uma vez que valores de *Dunn Index* maiores que 1 indicam resultados confiáveis.

A junção dos processos de mineração, juntamente com as métricas de validação, permite interpretar as características da rede (Seção 2.3) em relação à sua semântica no domínio do problema; os resultados relacionados serão discutidos na próxima seção.

3. Resultados

Esta seção apresenta e discute os resultados deste trabalho. Ela está dividida em duas partes; na Seção 3.1 são descritos os resultados da técnica de projeção multidimensional e na Seção 3.2 discute-se os que se referem a análise de agrupamentos.

3.1. Projeção Multidimensional

Em relação à população[3], a maioria das cidades do conjunto de dados é considerada como de pequeno porte, mas ainda apresenta um conjunto substancial de cidades de médio porte e um pequeno conjunto de cidades de grande porte, no qual São Paulo — a maior cidade brasileira — está localizada. Algumas análises preliminares podem ser feitas observando a Figura 1, onde as cidades (pontos) foram dimensionadas pela sua quantidade de vértices.

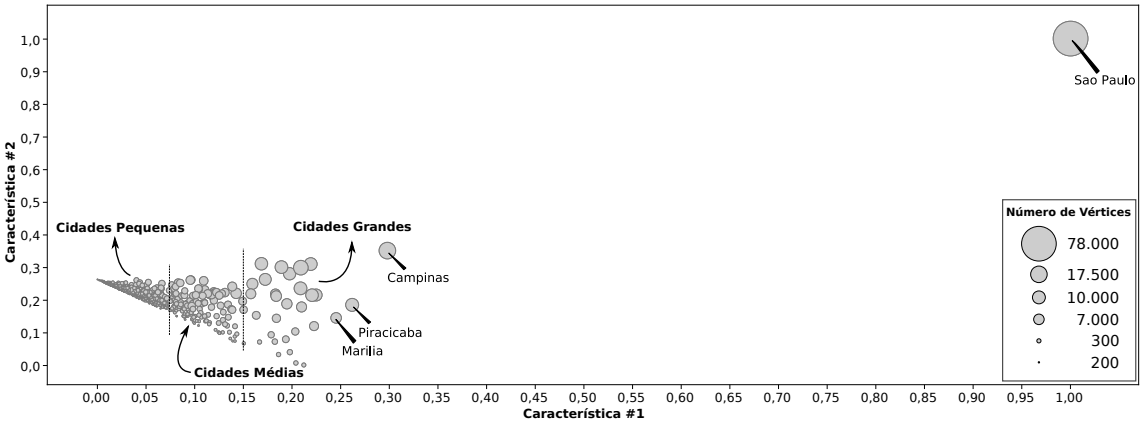


Figura 1: Representação dos vetores de características projetados em duas dimensões usando PCA; cidades são pontos dimensionados pelo número de vértices em suas redes.

Um indício de que as características topológicas selecionadas podem descrever conhecimento relevante sobre cidades é que a cidade de São Paulo está isolada das demais. Uma reação semelhante pode ser observada, em pequena escala, considerando as cidades de Campinas, Marília e Piracicaba, que estão separadas do grupo principal de cidades — localizado na parte esquerda da imagem. Acredita-se que esse comportamento está relacionado com a demografia das cidades; de modo que, em larga escala, as características topológicas podem inferir sobre a população das cidades, enquanto que, em pequena escala, podem apontar para bairros densamente ou escassamente povoados.

[3]Os detalhes sobre as categorias de tamanho são descritos nas Figuras 6a e 6b.

Na etapa subsequente, a cidade de São Paulo foi removida do conjunto de dados e o resultado desse processo foi ilustrado na Figura 2, considerando as técnicas de projeção PCA e Isomap, respectivamente; note que na imagem os valores foram normalizados no intervalo $[0, 1]$ a fim de facilitar a interpretação visual dos resultados.

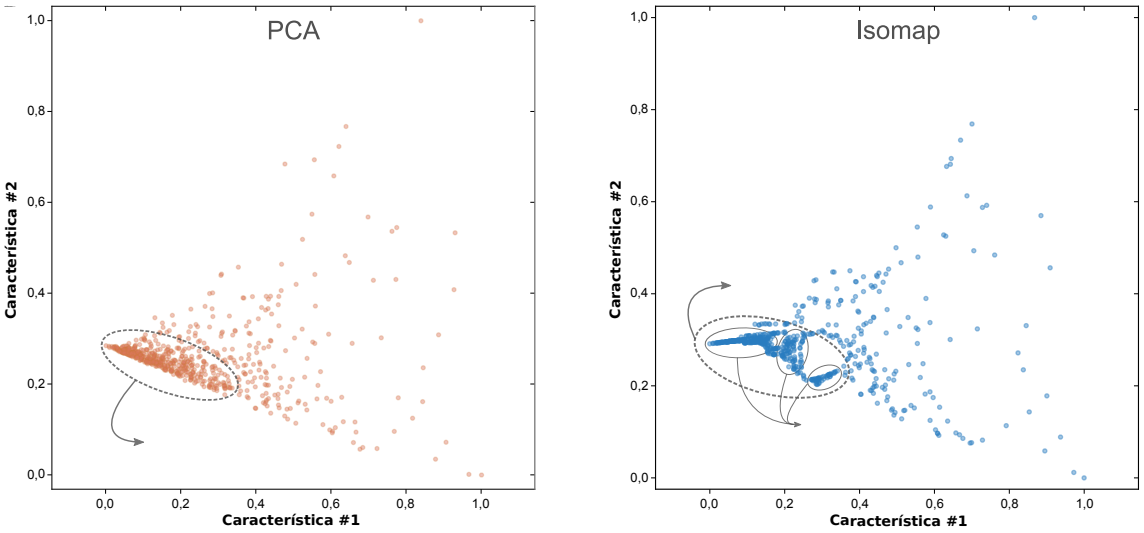


Figura 2: Projeção dos vetores de características das cidades usando PCA e Isomap.

As duas técnicas aplicadas mostram que os dados estão concentrados em uma pequena região de cada imagem, com poucos pontos esparsos ao longo dos eixos. A principal diferença entre as duas técnicas é que a Isomap implica na existência de múltiplas áreas com alta densidade, enquanto a PCA tem uma única área densa e muitos pontos esparsos. Isso é evidência de que cidades de pequeno porte tendem a se agrupar isolando cidades de médio e grande porte. Isomap, por outro lado, mostra que, apesar das cidades de pequeno porte serem semelhantes, elas têm particularidades que as divide em pequenos grupos dentro de um maior. Além disso, pode-se inferir que, por serem espalhadas, as cidades de tamanho médio e grande não possuem um padrão claro, mas ainda assim, elas podem compartilhar características comuns para serem exploradas pela análise de agrupamentos. Todavia, pode-se provar, usando correlação, que a topologia dos grafos pode inferir sobre a demografia das cidades do estado de São Paulo (ver Figure 3).

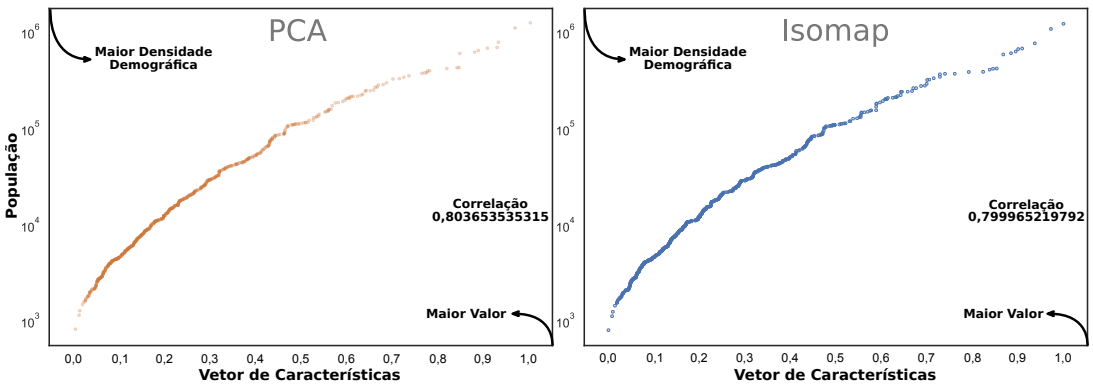


Figura 3: Teste de correlação entre o número de habitantes e as características das cidades projetadas em uma única dimensão usando ambas as técnicas PCA e Isomap.

Para avaliar a dependência entre os indicadores mediu-se sua correlação. Para esse fim, a dimensionalidade dos vetores de características foi reduzida para uma única dimensão. Sobre os valores resultantes, foram correlacionados os dados demográficos com as características unidimensionais de cada cidade. Como resultado, foi obtido 0,803 e 0,799 de correlação para PCA e Isomap, respectivamente. Ambos os valores indicam forte correlação entre os dados, permitindo afirmar que, no caso do estado brasileiro de São Paulo, as características topológicas e demográficas estão fortemente correlacionadas.

3.2. Análise de Agrupamentos

Na análise de agrupamentos utilizou-se KMeans variando o parâmetro da quantidade de grupos de 2 a 645 unidades, buscando pela configuração que proporciona a maior pontuação média de *Silhouette* (AVG) nos casos em que o *Dunn Index* (DNN) é maior do que 1. O resultado deste processo revelou que os dados são melhor agrupados em dois grupos, de modo que a cidade de São Paulo se separa das demais (ver Figuras 1 e 6c); este cenário possui os maiores valores de AVG e DNN, com AVG igual a 0,94 e DNN igual a 3,75 (ver Figura 4).

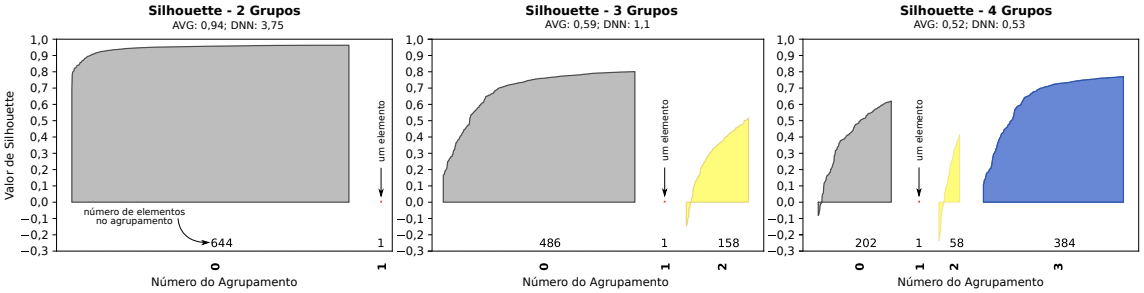


Figura 4: Avaliação da qualidade do agrupamento para todo o conjunto de dados.

Nota-se que, geralmente, a cidade de São Paulo está em um agrupamento isolado, enquanto que as outras cidades tendem a se agrupar, mesmo sendo geograficamente dispersas e notoriamente dissimilares umas às outras (ver Figuras 1 e 2). Ao remover São Paulo, é melhor dividir os dados em dois grupos (ver Figura 5), com os maiores valores de AVG e DNN sendo iguais a 0,59 e 1,10 respectivamente. Neste cenário, os agrupamentos aparentam um melhor equilíbrio quanto a quantidade de elementos por grupo, apontando para a existência de um padrão.

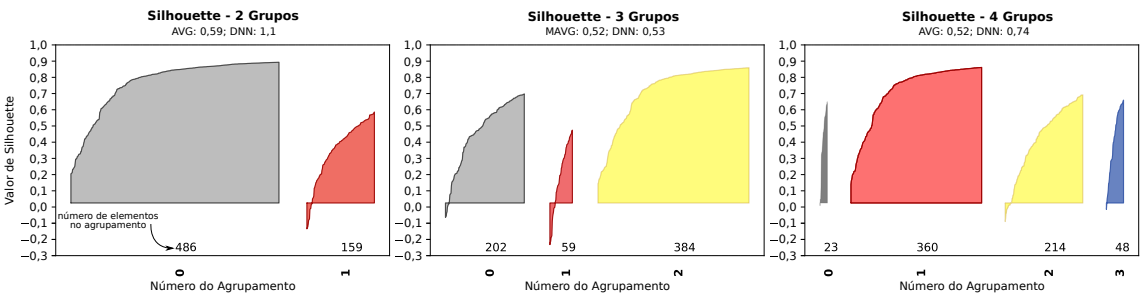
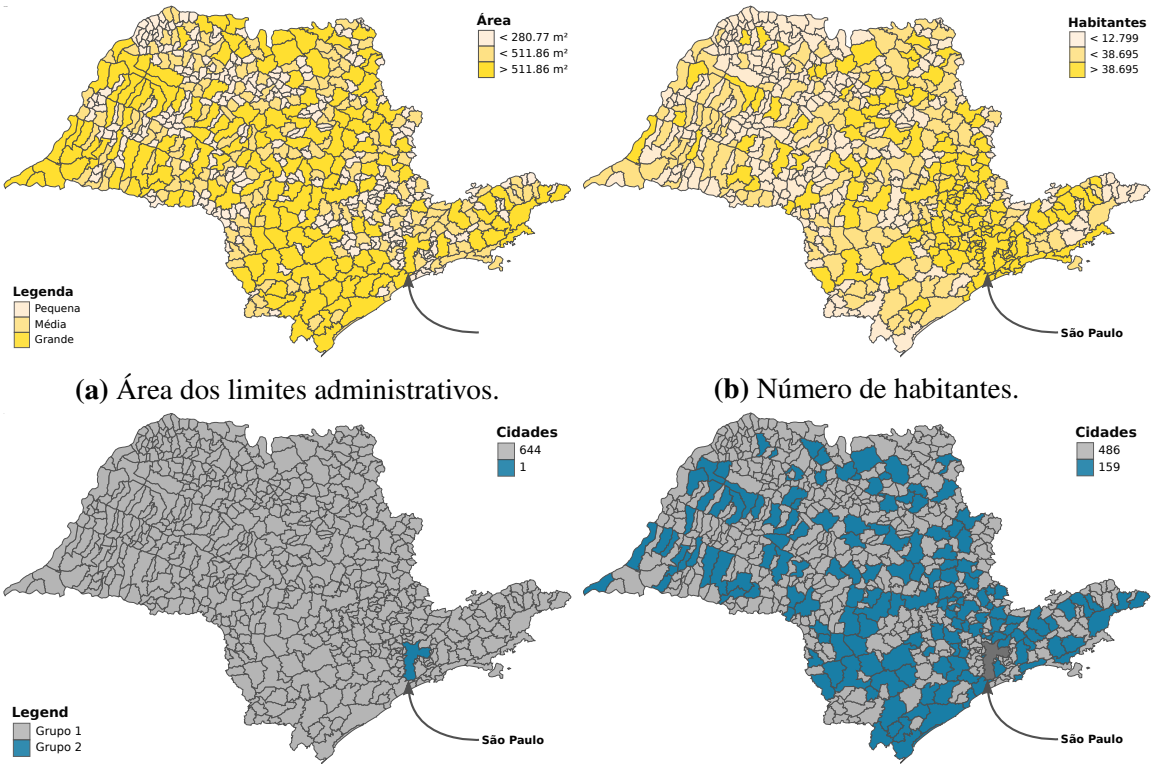


Figura 5: Avaliação da qualidade do agrupamento dos dados sem a cidade São Paulo.

Quanto ao último agrupamento, a relação que o favorece não está ligada ao número de habitantes, como a hipótese da seção anterior, mas sim à extensão territorial

(área em metros quadrados) de cada cidade. Observe que as características extraídas das redes estão conectadas a sua topologia a qual, por sua vez, está relacionada ao tamanho e geometria das malhas viárias. Além disso, 61,20% da população do estado está no primeiro grupo e 38,80% está no segundo (veja Figuras 5 e 6d). Não obstante, o segundo grupo parece ser povoado principalmente por cidades que são consideradas de extensão territorial média ou grande e apenas por algumas cidades pequenas (veja a Figura 6a).



(a) Área dos limites administrativos. **(b)** Número de habitantes.

(c) Agrupamento *com* São Paulo. **(d)** Agrupamento *sem* São Paulo.

Figura 6: Análise das cidades por meio de agrupamentos e indicadores urbanos relacionados à área dentro de seus limites administrativos e ao seu número de habitantes.

Esse padrão pode ser entendido como a forma com que as cidades se organizam em seu espaço disponível. De fato, em relação à extensão territorial, 61,54% das cidades do primeiro grupo são de tamanho pequeno, 25,78% são de tamanho médio e 12,58% são de tamanho grande; enquanto que 13,91% das do segundo grupo são de tamanho pequeno, 22,78% são de tamanho médio e 63,29% são de tamanho grande. Portanto, conclui-se que em relação a quantidade populacional as cidades do primeiro grupo podem ser consideradas pequenas e densamente povoadas, enquanto que as do segundo grupo podem ser consideradas grandes e escassamente povoadas.

4. Conclusão

Neste trabalho, foram analisadas características extraídas de 645 cidades que formam o estado de São Paulo. A metodologia aplicada baseou-se em processos de mineração de dados, com foco em projeção multidimensional e análise de agrupamentos, dividindo-se em processos de (i) Aquisição e Preparação de Dados, (ii) Extração e Seleção de Características, e (iii) Análise de Vetores de Características. Os resultados descrevem as

relações entre redes viárias, sua demografia e extensão territorial, explicando associações entre a topologia e indicadores urbanos das cidades. Mais precisamente, as contribuições deste trabalho estão na descrição de como a topologia da rede é capaz de revelar grupos de cidades com características semelhantes, na análise de correlação entre a quantidade de população das cidades e suas características, e no estudo de porquê as cidades se agrupam com outras distantes e não com aquelas as quais fazem fronteira.

Agradecimentos

Os autores são gratos ao CNPq (167967/2017-7), a FAPESP (2016/17078-0, 2016/17330-1 e 2017/08376-0) e a CAPES (10095541/M) pelo apoio financeiro a este trabalho.

Referências

Anderson, T. K. (2009). Kernel density estimation and k-means clustering to profile road accident hotspots. *Accident Analysis & Prevention*, 41(3):359–364.

Blumer, H. (1971). Social problems as collective behavior. *Social problems*, 18(3):298–306.

Boccaletti, S., Latora, V., Moreno, Y., Chavez, M., and Hwang, D. (2006). Complex networks: Structure and dynamics. *Physics Reports*, 424(4-5):175–308.

Chiang, C. (2003). *Statistical Methods of Analysis*. World Scientific.

Costa, L. F., Rodrigues, F. A., Travieso, G., and Boas, P. R. V. (2007). Characterization of complex networks: A survey of measurements. *Advances in Physics*, 56(1):167–242.

Costa, L. F., Travençolo, B. A. N., Viana, M. P., and Strano, E. (2010). On the efficiency of transportation systems in large cities. *EPL (Europhysics Letters)*, 91(1).

Crucitti, P., Latora, V., and Porta, S. (2006). Centrality measures in spatial networks of urban streets. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 73(3).

Domingues, G. S., Silva, F. N., Comin, C. H., and Costa, L. F. (2017). Topological characterization of world cities. *arXiv preprint arXiv:1709.08244*.

Grauwin, S., Sobolevsky, S., Moritz, S., Gódor, I., and Ratti, C. (2015). Towards a comparative science of cities: Using mobile traffic records in new york, london, and hong kong. In *Computational approaches for urban environments*, pages 363–387. Springer International Publishing.

Konstantopoulos, T. (2012). *Introduction to projective geometry*. Number September. Dover Publications.

Kremer, H., Kranen, P., Jansen, T., Seidl, T., Bifet, A., Holmes, G., and Pfahringer, B. (2011). An effective evaluation measure for clustering on evolving data streams. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 868–876, New York, NY, USA. ACM.

Li, X. and Parrott, L. (2016). An improved genetic algorithm for spatial optimization of multi-objective and multi-site land use allocation. *Computers, Environment and Urban Systems*, 59:184–194.

MacQueen, J. et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297.

Masucci, A. P., Stanilov, K., and Batty, M. (2013). Limited Urban Growth: London's Street Network Dynamics since the 18th Century. *PLoS ONE*, 8(8).

Pan, G., Qi, G., Zhang, W., Li, S., Wu, Z., and Yang, L. T. (2013). Trace analysis and mining for smart cities: issues, methods, and applications. *IEEE Communications Magazine*, 51(6):120–126.

Porta, S., Crucitti, P., and Latora, V. (2006a). The network analysis of urban streets: A dual approach. *Physica A: Statistical Mechanics and its Applications*, 369(2):853–866.

Porta, S., Crucitti, P., and Latora, V. (2006b). The Network Analysis of Urban Streets: A Primal Approach. *Environment and Planning B: Planning and Design*, 33(5):705–725.

Porta, S., Latora, V., Wang, F., Strano, E., Cardillo, A., Scellato, S., Iacoviello, V., and Messora, R. (2009). Street Centrality and Densities of Retail and Services in Bologna, Italy. *Environment and Planning B: Planning and Design*, 36(3):450–465.

Scripps, J., Nussbaum, R., Tan, P.-N., and Esfahanian, A.-H. (2010). Link-Based Network Mining. In *Structural Analysis of Complex Networks*, pages 403–419. Springer Nature.

Spadon, G., Gimenes, G., and Rodrigues-Jr, J. F. (2017). Identifying Urban Inconsistencies via Street Networks. volume 108, pages 18 – 27. Elsevier BV. International Conference on Computational Science, ICCS 2017, 12-14 June 2017, Zurich, Switzerland.

Spiwok, V., Oborsky, P., Pazurikova, J., Keenek, A., and Kralova, B. (2015). Nonlinear vs. linear biasing in trp-cage folding simulations. *The Journal of Chemical Physics*, 142(11).

Strano, E., Nicosia, V., Latora, V., Porta, S., and Barthélemy, M. (2012). Elementary processes governing the evolution of road networks. *Scientific Reports*, 2.

Strano, E., Viana, M., Costa, L. F., Cardillo, A., Porta, S., and Latora, V. (2013). Urban Street Networks, a Comparative Analysis of Ten European Cities. *Environment and Planning B: Planning and Design*, 40(6):1071–1086.

Workload-aware Parameter Selection and Performance Prediction for In-memory Databases

Maria I. V. Lima[1], **Victor A. E. de Farias**[1],
Francisco D. B. S. Praciano[1], **Javam C. Machado**[1]

[1]Laboratório de Sistemas e Bancos de Dados (LSBD)
Computer Science Dept – UFC – CEP 60440-900 – Fortaleza – CE – Brazil

{isabel.lima,victor.farias,daniel.praciano,javam.machado}@lsbd.ufc.br

Abstract. *In-memory databases, just as hard drive ones, may offer hundreds of customizable settings, making the task of system tuning overwhelming for a database administrator. Even worse, the number of parameters continues to grow over the years and they can affect performance in a not intuitive manner. Models that capture their behavior can assist automatic tuning mechanisms to obtain optimal performance. In this work, we propose a learning-based approach to select the most meaningful parameters and generate a performance model based on both the workload and the database configurations. Experimental results confirm that our approach can create accurate performance models using only a reduced set of selected parameters.*

1. Introduction

For a long time, conventional disk resident databases were the dominant storage technology in the market. Over the years, however, the prices of physical memory have significantly decreased as shown in figure 1, while its storage capacity has increased, thus making it affordable to have whole databases fit in main memory. This is, in fact, the precise definition of an in-memory database: one where data resides permanently in memory [Garcia-Molina and Salem 1992].

The growth of in-memory databases has made it possible to achieve great performance gain, as they can reportedly be up to 50,000 times faster than disk-based systems [Lake and Crowther 2013]. This is greatly due to the fact that, comparatively, there is much less disk input/output (I/O) latency (which is known to be the biggest bottleneck in databases) being introduced in these systems. At the same time, the design of an in-memory system is very different as it has to be optimized to take advantage of memory use. Such differences may include, for instance, the absence of buffer management, use of alternative indexing structures such as T-Trees and Bw-Trees [Levandoski et al. 2013], and use of latches instead of locks. This leads to the questions of whether and how these architectural changes may reflect in the evaluation of system performance. What different aspects may become a bottleneck for such systems?

One insightful aspect to be explored regarding performance is configuration. Today's software systems are extensively customizable, often with hundreds of configuration parameters. This is true for different kinds of systems and database systems are no exception. Not only there are too many settings, but also they are constantly being renamed, excluded and added, making it difficult for a database administrator (DBA) to

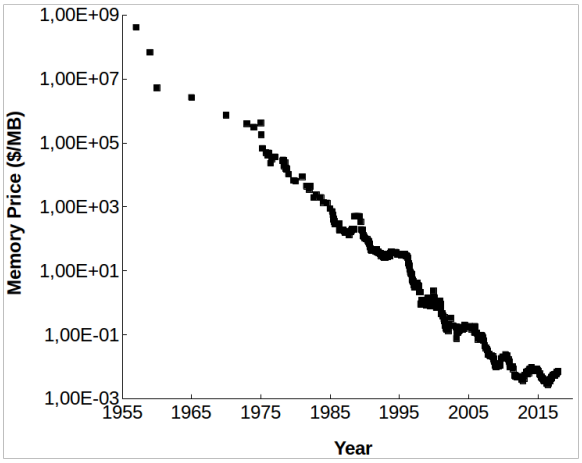


Figure 1. Cost of main memory with time [McCallum 2017]

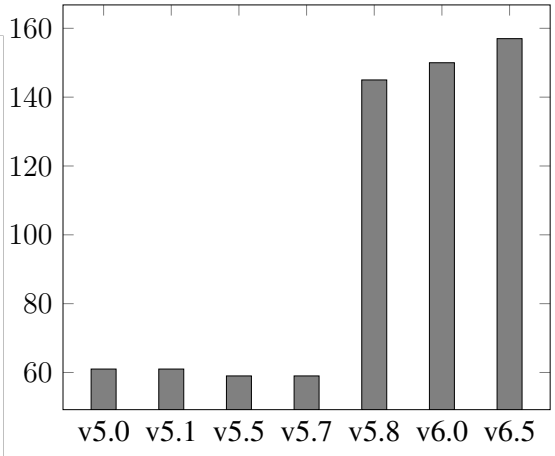


Figure 2. Number of configuration parameters in different versions of MemSQL

keep track of them all. Figure 2 shows some of these statistics for the MemSQL database system [Shamgunov 2014], where we can see how the number of configuration parameters grows over the released versions. On top of this, there are plenty of settings that have dependencies between each other, or that do not affect performance in a linear or intuitive way, or even that do not necessarily require management, i.e., they present little or no direct impact to the applications. In short, the amount of configuration parameters in modern databases makes tuning them a complex and time-consuming task.

This paper presents the results of an investigation of the impact of parameter configuration in in-memory database systems. Its contributions are:

- Identifying the minimum set of most impacting parameters, thus reducing the complexity of managing these systems;
- Analyzing the selected parameters individually and further discussing the underlying reasons for their level of importance in the context of in-memory databases;
- Providing a machine learning model for performance prediction based on the subset of selected parameters that captures the non-linear behaviour of the configuration parameters.

We model the performance of an in-memory database using two meta-parameters: workload configuration and database configuration. Workload configuration comprises a set of parameters that define a template for a workload, composed of a mix of transactions, while database configuration is defined by a collection of internal database settings, often referred to as “knobs”, provided by a manufacturer. The combination of these two meta-parameters, as it will be shown in experiments, is sufficient to provide a solid indicator of performance, whichever metric is being used to evaluate it.

2. Related Work

Several works have addressed the question of how to analyze the influence of workload and configuration parameters on the overall database performance. One way to do this is to employ statistical models so that it is possible to understand the impact of both

the configuration and the workload, thus enabling a more sophisticated analysis that can predict performance.

Ganapathi et al. (2009) proposed an approach to build a model that is capable of predicting the performance metrics accurately. To do so, the authors employ a variety of statistical machine learning techniques. In especial, they develop a model using a modified version of Kernel Canonical Correlation Analysis (KCCA). The evaluation of this model claims that it is able to predict the performance metrics in an accurate way, but it suffers from a few limitations. The main one is that the model aims to predict the performance metrics of specific queries only.

On the opposite and complementary way to the work mentioned above, Mozafari et al. (2013) explored two kinds of models, denominated black-box and white-box, that are able to predict the performance metrics. The black-box models make minimal assumptions about the context where they will be applied, whereas the white-box models make several assumptions. For the the black-box models, the authors present several machine learning regression techniques that can be used, such as KCCA and Decision Trees. In the context of white-box models, they analyze MySQL features (e.g., 2-phase locking algorithm, buffer replacement policies, etc.) to build an accurate cost-based model for the database. The results demonstrated that both of these models are able to predict the maximum throughput with relative errors within $0 - 25\%$ for a OLTP workload. It is worthy to note that none of the proposed models take into account the current settings of the database, i.e., they are solely based on SQL query logs and OS statistics that was previously collected.

Another approach is to combine different machine learning models in order to perform a feature selection before the performance analysis, because it is often hard to deal with the very large number of configuration parameters available to be adjusted in modern databases. Furthermore, evidence suggests that not all of the available settings, often in the order of hundreds, necessarily need to be tuned for a good performance [Xu et al. 2015]. A number of recent works have used this approach to develop tools and frameworks for automatically optimizing database parameters.

In order to develop a tuning tool known as iTuned, Duan et al. (2009) has proposed an approach that uses an intermediate step that is responsible for making a feature selection in order to find the highest-impact parameters. To do that, the authors employ the Statistical Approach for Ranking Database Parameters (SARD) [Debnath et al. 2008] combined with another technique denominated Adaptive Sampling. This approach is able to select the most important parameters, but one of its limitations is that it does not consider the database parameters during the feature selection process.

Likewise to the above paper, OtterTune [Aken et al. 2017] is another tool that applies feature selection as a part of a process that aims to suggests optimal configurations based on previously seen workloads and collected metrics using clusterization techniques. To do so, the authors used Factor Analysis (FA) and k-means in order to select and cluster the most important metrics. Moreover, the Lasso regression method [Tibshirani 1996] is employed to rank the knobs. Using this method, OtterTune effectively selects the proper knobs, but its focus is on the recommendation of configurations rather than on prediction.

Objectively comparing our work with all of those cited above, we observe that

none of them involve using the values of internal database parameters in the construction of the model to predict the performance indicators. Hence, the main differences from them are that i) we focus on in-memory databases, employing one in our experiments and discussing the results in light of its particularities and ii) this work aims to provide a model in which the performance is based on both the database parameters and the workload.

A comparison between the main aspects of the different works discussed in this section is shown in table 1.

Work	Database Type	Model Input	Prediction	Feature Selection
[Ganapathi et al. 2009]	Disk-based	Workload	Yes	No
[Mozafari et al. 2013]	Disk-based	Workload	Yes	No
[Duan et al. 2009]	Disk-based	Workload	No	Yes
[Aken et al. 2017]	Disk-based	Workload	No	Yes
This work	In-memory	Settings and workload	Yes	Yes

Table 1. Summary of related works

3. Our Approach

Our approach strives to select the minimum set of most impacting configuration parameters of an in-memory database while creating performance models for estimating performance metrics. We propose a data-driven machine learning-enabled strategy that employs a wrapper feature selection technique, i.e., it uses supervised learning methods for assessing the quality of subsets of configuration parameters and thus selecting the best subset.

Supervised learning is able to approximate functions by generalizing the behavior of examples. It requires a dataset composed by samples of the independent input variables (database and workload configurations) and their corresponding dependent output variable value (performance metric). Therefore, experiments are executed to generate a performance dataset to feed the learning algorithm.

In order to facilitate the understanding of the proposed solution, we divide this section in i) Dataset generation; ii) Feature selection and iii) Performance prediction.

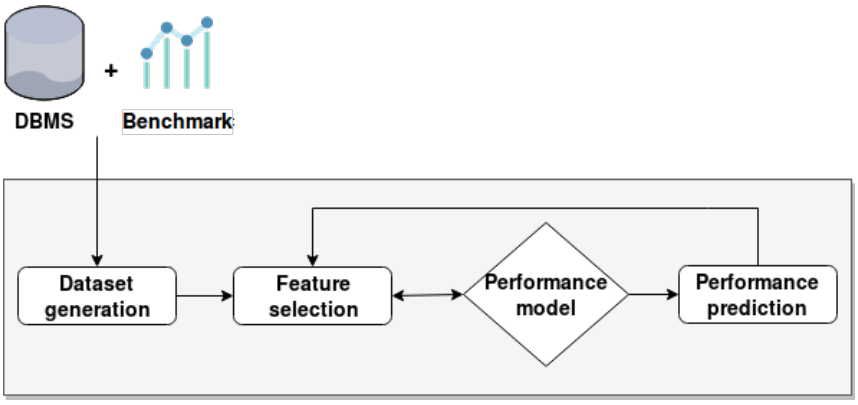


Figure 3. Approach outline

3.1. Dataset generation

Performance is dependent on both workload and database configurations (as shown in equation 1). Thus we generate a representative dataset that covers i) the domain of each configuration parameter to learn their individual influence on the performance and ii) the several combinations of variables values to expose dependencies between variables.

$$Performance = f(Config_{Database}, Config_{Workload}) \quad (1)$$

To create the performance dataset, we carry out experiments on a test environment which is a copy of the production environment. In these experiments, we vary the database and the workload configurations. The workload configuration ($Config_{Workload}$) is represented by the frequencies $< f_1, f_2, \ldots, f_t >$ where t is the number of workload templates and f_i is the frequency of the i-th workload template. For instance, in the TPC-C benchmark [The Transaction Processing Council 2007] there are five workload templates, and their combination generates different workload configurations.

The database configuration ($Config_{Database}$) comprises a group of parameters available to be adjusted by the database manufacturer. Some of these can only be set during the database creation, while others can be modified with every connection. Such details are particular to each manufacturer, with different database systems offering different sets of configuration parameters, but most of them share a great number of common parameters, even if they are not labeled the same. It is represented by $< p_1, p_2, ..., p_k >$, where k is the number of configuration parameters and p_i is the value of the i-th configuration parameter. Note that these values can be either continuous, discrete or categorical.

Thus, we execute 3,000 experiments, where each one of them corresponds to one entry in the dataset. In each experiment, random values are generated for each workload template and each database configuration parameter within their own domains, in order to produce different combinations of settings. Each experiment runs for two minutes where the first 20 seconds of warm-up and the last 20 seconds of cool-down are removed. A set of predefined performance metrics, such as latency and throughput, are collected during this process. This is represented by $< m_1, m_2, \ldots, m_j >$, where j is the number of metrics collected and m_i is the value of the i-th performance metric. In the end, the generated dataset has the following structure:

	Workload templates	Configuration parameters	Performance metrics
1	$< f_1, f_2, \ldots, f_t >$	$< p_1, p_2, \ldots, p_k >$	$< m_1, m_2, \ldots, m_j >$
...	...	...	...
n	$< f_1, f_2, \ldots, f_t >$	$< p_1, p_2, \ldots, p_k >$	$< m_1, m_2, \ldots, m_j >$

3.2. Feature selection

In this phase, we aim to eliminate the less meaningful database configuration parameters, the ones that produce little or no impact in the performance. We rely on a feature selection method that captures the most important variables based on a certain performance metric.

We employ Recursive Feature Elimination (RFE) [Guyon et al. 2002], which iteratively reduces the set of database parameters by wrapping a supervised learning method

to assess the quality of a subset of parameters. This supervised learning algorithm plays two roles in this process: i) evaluate the quality of a set of parameters by assessing their ability to predict the performance metric, as described in subsection 3.3, and ii) rank the importance of the variables, using methods like linear regression, that assigns weights to variables, or decision tree, that computes Gini coefficients for each variable.

This algorithm first considers the set of all features (parameters). In each round, it assesses the quality of this set of parameters by training a supervised model and computing its accuracy metric. Then, the trained model ranks the variables according to the weights assigned to them. As greater weights have more influence on the model, the last ranked parameter is removed and the model is re-trained using the remaining features in the next round. This process continues repeatedly until it has no remaining parameters left to test, as they have all been eliminated. The selected parameters are the ones in the round that yielded the best value for the accuracy metric.

Next, we demonstrate how the dataset referenced in section 3.1 is used in this phase. First, a performance metric m_i is chosen from the dataset as the target value, depending on the application requirements or the DBA's preference. Then, the examples (i.e. each line of the dataset) are used in the RFE algorithm to train a model using a regressor as an estimator. The attributes of the examples are the workload templates and the database configuration parameters, and the target value is the chosen metric. Each round of RFE iteratively eliminates one of these attributes.

3.3. Performance prediction

The performance prediction phase constructs models that can estimate performance metrics based on the values of the database and workload parameters. Note that the performance metrics may have a non-linear relationship with database parameters [Aken et al. 2017] and to its workload parameters [Mozafari et al. 2013]. Additionally, two parameters can show dependency [Aken et al. 2017] to each other. Predictive models should capture this behavior in order to deliver accurate estimates. This fact lead us to choose non-linear methods instead of linear models.

Non-linear models also measure the quality of a subset of features. By training a model with a given subset of variables, we can compare the prediction power of this subset to the whole set of variables. If using subset is as accurate as the whole subset, it means that the remaining variables do not account for the target value. Furthermore, in our experiments, we show that models trained with a subset of variables can be more accurate than using the whole set of variables.

As performance metrics, we use throughput and latency. Predicting these metrics is a supervised learning problem and, specifically, it is a regression task since their values are discrete and unbounded. In this work, we test several regression methods in order to evaluate which of them is more accurate for our problem.

We evaluate the quality of a model in terms of accuracy metrics. We employ the k-fold technique [Stone 1974] along with the mean absolute error (MAE) metric. k-fold computes the accuracy while being robust to both under and overfitting. k-fold divides the whole dataset equally in k folds. For each fold, it uses this fold as test dataset to compute MAE and uses the remaining $k - 1$ folds as train dataset. At the end, we take the mean value of MAE for each fold. This value is used to compare the accuracy between methods

and for evaluating subsets of parameters in the RFE algorithm. Additionally, choosing the right parameters to address can help regression methods to improve accuracy.

The generated model is able to receive a certain database and workload configuration as input and predict its performance, based on the previously seen data used to train it. Using it will save time from directly executing a workload under a given configuration to discover how long it takes to execute, for example.

4. Experimental Evaluation

In this section, we aim to evaluate our proposed approach through a series of experiments. In particular, it is our intention to show that our strategy:

- chooses the best subset of features that directly impact on performance by employing RFE, where the accuracy of predictive models is used to assess the quality of parameter subsets. The accuracy is measured using k-fold along with MAE;
- predicts performance metrics based solely on the workload and database configurations;
- provides insights for understanding the underlying mechanisms of how the configuration parameters changes the in-memory system behavior. We also provide an analysis of the main aspects behind the parameters outputted by the feature selection phase.

4.1. Experimental Setup

We relied on a commercial in-memory database that will be referred to as database A. All tests were executed in a machine with $32GB$ of RAM and 4 cores running on Ubuntu 14.04.5 LTS. For generating workloads and collecting performance metrics, we used TPC-C benchmark implemented in OLTP-Bench [Difallah et al. 2013]. While running TPC-C, a scale factor of 120 (equivalent to a database of roughly $18GB$) and a time window of two minutes of benchmark execution were used. All implementations were written in Python with intense use of the $scikit - learn$ library [Pedregosa et al. 2011] for machine learning algorithms.

4.2. Result Analysis

We tested the RFE algorithm using three different machine learning techniques: Gradient Boosting Machine (GBM) [Friedman 2001], Random Forest (RF) [Breiman 2001] and Decision Tree (DT) [Breiman et al. 1984]. In figure 4, the evolution of the applied RFE algorithm along with the number of iterations, as features are eliminated, is shown for each of the three different methods. This graphic shows that all methods eliminated the right features at the beginning of the run since MAE raises only with less than about 12 features. The difference between them is the minimum MAE and the number of features with minimum MAE.

Thus table 2 displays the minimum MAE, which is equivalent to the optimal point, obtained in each method, along with the respective number of selected features. We note that the Gradient Boosting Machine method outperformed Random Forest and Decision Tree on both MAE and the number of selected features. It selected the smaller number of features with the smaller MAE.

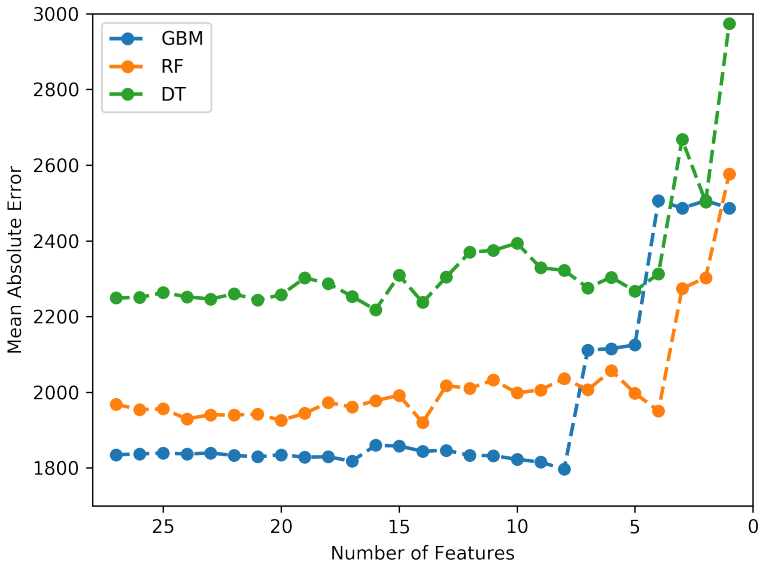


Figure 4. RFE iterations: MAE vs. Number of Features

In addition, this information is complemented by tables 3 to 5. They show, for each of the three methods, the name of the selected parameters (and their respective score) for the RFE round that yielded the best MAE. Random Forest, for example, generated a minimum MAE of 1919.7 and selected 11 parameters, which are listed in table 4 sorted by decreasing order of importance. These scores are the weights assigned to each feature by training the estimator method (GBM, RF or DT). Many parameters, like Commit Durability, Transaction Log File Size and Transaction Log Buffer Size, were selected by all methods, showing consistency between them.

	Gradient Boosting	**Random Forest**	**Decision Tree**
Min. MAE	1796.56	1919.7	2217.59
Number of Features	6	11	12

Table 2. Comparison of minimum MAE and number of selected features using different algorithms

Parameter	**Score**
Checkpoint Log Volume	0.164911
Transaction Log Buffer Size	0.147690
Transaction Log File Size	0.135332
Max. Commit Buffer Size	0.130672
Checkpoint Rate	0.128974
Commit Durability	0.119779

Table 3. Parameters selected by RFE using Gradient Boosting Machine

Next, we choose some of the selected configuration parameters shown in tables 3 to 5 to discuss about. The focus of this discussion is to better understand the importance

Parameter	Score
Commit Durability	0.212208
Lock Level	0.149018
PL/SQL Memory Size	0.062042
SQL Query Timeout	0.053479
Transaction Log Buffer Size	0.053343
Lock Wait Time	0.052239
PL/SQL Conn. Mem. Limit	0.049722
Checkpoint Log Volume	0.048706
Transaction Log File Size	0.047432
Max. Commit Buffer Size	0.046462
Checkpoint Frequency	0.042661

Table 4. Parameters selected by RFE using Random Forest

Parameter	Score
Commit Durability	0.230500
Lock Level	0.156808
SQL Query Timeout	0.058409
PL/SQL Conn. Mem. Limit	0.053595
Checkpoint Rate	0.053550
Log Buffer Parallelism	0.048701
Transaction Log File Size	0.045762
Checkpoint Log Volume	0.041598
Checkpoint Frequency	0.035915
Transaction Log Buffer Size	0.034257
Log Purge	0.029058
Max. Commit Buffer Size	0.023990

Table 5. Parameters selected by RFE using Decision Tree

of these aspects in the context of in-memory databases, bringing an insight to the reasons behind their selection.

- **Commit Durability:** Whenever a transaction is committed, its log record may or may not be immediately written to disk. Writing the transaction log to disk right after a commit means that no committed transactions will be lost in case of failure. However, this benefit comes at the expense of a lengthier execution due to the added disk access time.
- **Checkpoint Log Volume:** Checkpoints occur with a certain frequency and, between two consecutive ones, the amount of data that can be stored in the log file may be limited. If the log file is full before the next checkpoint happens, new data will not be written to the log, resulting in its loss. Thus it is important to make sure that the parameter that controls this aspect is reasonably set.
- **Transaction Log File Size:** Specifying a maximum file size for the transaction log is likely to be a bottleneck-inducing concern when tuning a database. This aspect may present an issue in the case of unknowingly having the log file being set too small, causing constant checkpoint operations (which are costly and heavily I/O-bound) every time the log file reaches its maximum size.
- **Transaction Log Buffer size:** Before being written to disk, data is generally stored in buffers in main memory. For example, it is common to have transaction commit and log buffers in database systems. Setting a buffer size too small may cause the buffer to get full very quickly, thus forcing it to recurrently flush its data to disk. Therefore, making sure that the buffers are large enough to avoid the overhead of constant disk access is an important precaution.
- **Max. Commit Buffer Size:** Similar to what happens to Transaction Log Buffer Size, limiting the size of the transaction commit buffer may cause constant disk flushing. It is good practice to make sure that the commit buffer size is large enough to avoid this happening too often.
- **Checkpoint Rate:** As checkpoint is an operation that makes use of disk access, setting the rate at which they are written to disk too low may cause the checkpoint

process to take a lot longer than expected. This may as well affect other operations, causing delays in backup and recovery times.

- **Lock Level:** This is a potentially critical parameter, as setting the lock to a not suitable level may undermine the database performance. Setting a database-level lock may significantly slow down the access of transactions. Generally speaking, row-level locking or even table-level locking tend to be better solutions to maximize concurrency control, but this is not always the case.
- **Lock Wait Time:** Deciding a suitable lock wait time acts as a double-edged sword. On the one hand, having this parameter set too high may cause transactions to wait an unnecessarily long time until they are able to obtain the lock. On the other, setting it too low will cause a great number of transactions to be aborted early, making it important to keep an eye for a balance regarding this parameter.

Ultimately, even though in-memory databases are much faster than hard disk ones due to them relying on main memory access, disk access still counts as a considerable part of the mechanics of these systems. It is possible to infer from the experiments that concurrency control also plays an important role in their performance.

The use of inadequate levels and other locking-related issues should, for sure, be carefully watched. But, in the end, disk I/O continues to be the utmost concern when looking to speed up execution times, because even in-memory databases still need some sort of persistence due to main memory's volatility. Writing and reading log files and making periodic system checkpoints are common database maintenance operations that heavily depend on I/O and inevitably introduce undesirable latency.

We note that, depending on the database used for the experiments and the configuration parameters provided, or on the generated dataset used to perform the feature selection, the results could differ accordingly. However, this approach is generic and can be applied to databases of any kind (both in-memory and disk-based). Also, it is relatively cheap to build the performance model. Once the dataset is generated (which is the most time-consuming phase), training the model can be done rather quickly.

Lastly, we assess the reduction power of feature selection. Considering that the number of original parameters in the experiments was 29, our approach achieved a reduction of 58.62% for DT, 62.06% for RF and 79.3% for GBM in the number of parameters.

5. Conclusion and Future Work

In this work, we have addressed the problem of building a performance model that takes into account both workload and database configurations. We show that it is possible to, under a given workload, reduce a potentially large set of database parameters to a smaller subset that impacts the most a chosen performance indicator. This allows database administrators to save time during the process of system tuning.

In specific, our experiments show that the use of RFE with Gradient Boosting Machine yielded the best results for the feature selection. It presented the smallest error between the three tested methods, resulting in a subset of only 6 parameters, which is a reduction of dozens of parameters.

We also analyze some of the most performance-impacting aspects regarding in-memory databases pointed out by our experiments. Upon the presented results, we con-

clude on how disk access still plays a great part in the performance of these systems, accounting for multiple possible bottleneck situations. Therefore, we highlight the importance of understanding configuration parameters and properly tuning them.

To add more supporting evidence to this work, a future extension is carrying out similar experiments to those presented here, but using a variety of different benchmarks (e.g. YCSB, TPC-H, etc.) and in-memory databases (e.g. VoltDB [Stonebraker and Weisberg 2013] or Peloton [Pavlo et al. 2017]).

Another possible future contribution is using active learning to build a precise database performance model more quickly. Instead of randomly choosing workload and database configurations in an attempt to cover the extensive search space, active learning would suggest which is the next best data point (configuration) to be labeled, thus saving time from labeling less meaningful cases.

Lastly, we plan to extend our work to automatically tune the parameters to optimize system's performance. Black-box optimization methods may be suited for this case since we do not possess the closed form of the performance function.

Acknowledgments

This research was partially funded by CAPES (grants #1697978 and #1782887) and LSBD/UFC.

References

Aken, D. V., Pavlo, A., Gordon, G. J., and Zhang, B. (2017). Automatic Database Management System Tuning Through Large-scale Machine Learning. In *Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD Conference 2017, Chicago, IL, USA, May 14-19, 2017*, pages 1009–1024.

Breiman, L. (2001). Random Forests. *Machine Learning*, 45(1):5–32.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees*. Wadsworth.

Debnath, B. K., Lilja, D. J., and Mokbel, M. F. (2008). SARD: A statistical approach for ranking database tuning parameters. In *Proceedings of the 24th International Conference on Data Engineering Workshops, ICDE 2008, April 7-12, 2008, Cancún, México*, pages 11–18.

Difallah, D. E., Pavlo, A., Curino, C., and Cudré-Mauroux, P. (2013). OLTP-Bench: An Extensible Testbed for Benchmarking Relational Databases. *PVLDB*, 7(4):277–288.

Duan, S., Thummala, V., and Babu, S. (2009). Tuning Database Configuration Parameters with iTuned. *PVLDB*, 2(1):1246–1257.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

Ganapathi, A., Kuno, H. A., Dayal, U., Wiener, J. L., Fox, A., Jordan, M. I., and Patterson, D. A. (2009). Predicting Multiple Metrics for Queries: Better Decisions Enabled by Machine Learning. In *Proceedings of the 25th International Conference on Data Engineering, ICDE 2009, March 29 2009 - April 2 2009, Shanghai, China*, pages 592–603.

Garcia-Molina, H. and Salem, K. (1992). Main Memory Database Systems: An Overview. *IEEE Trans. Knowl. Data Eng.*, 4(6):509–516.

Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422.

Lake, P. and Crowther, P. (2013). In-memory databases. In *Concise Guide to Databases*, pages 183–197. Springer.

Levandoski, J. J., Lomet, D. B., and Sengupta, S. (2013). The Bw-Tree: A B-tree for new hardware platforms. In *29th IEEE International Conference on Data Engineering, ICDE 2013, Brisbane, Australia, April 8-12, 2013*, pages 302–313.

McCallum, J. C. (2017). Memory prices (1957-2017). `https://jcmit.net/memoryprice.htm`. Accessed: 2018-03-05.

Mozafari, B., Curino, C., Jindal, A., and Madden, S. (2013). Performance and resource modeling in highly-concurrent OLTP workloads. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2013, New York, NY, USA, June 22-27, 2013*, pages 301–312.

Pavlo, A., Angulo, G., Arulraj, J., Lin, H., Lin, J., Ma, L., Menon, P., Mowry, T. C., Perron, M., Quah, I., Santurkar, S., Tomasic, A., Toor, S., Aken, D. V., Wang, Z., Wu, Y., Xian, R., and Zhang, T. (2017). Self-Driving Database Management Systems. In *CIDR 2017, 8th Biennial Conference on Innovative Data Systems Research, Chaminade, CA, USA, January 8-11, 2017, Online Proceedings*.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., VanderPlas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Shamgunov, N. (2014). The MemSQL In-Memory Database System. In *Proceedings of the 2nd International Workshop on In Memory Data Management and Analytics, IMDM 2014, Hangzhou, China, September 1, 2014*.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions. *Journal of the royal statistical society. Series B (Methodological)*, pages 111–147.

Stonebraker, M. and Weisberg, A. (2013). The VoltDB Main Memory DBMS. *IEEE Data Eng. Bull.*, 36(2):21–27.

The Transaction Processing Council (2007). TPC-C Benchmark (Revision 5.11). `http://www.tpc.org/TPC_Documents_Current_Versions/pdf/tpc-c_v5.11.0.pdf`.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288.

Xu, T., Jin, L., Fan, X., Zhou, Y., Pasupathy, S., and Talwadker, R. (2015). Hey, you have given me too many knobs!: understanding and dealing with over-designed configuration in system software. In *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering, ESEC/FSE 2015, Bergamo, Italy, August 30 - September 4, 2015*, pages 307–319.

Database Tuning with Partial Indexes

Alain D. Fuentes[1], Ana Carolina Almeida[2], Rogério Luís de Carvalho Costa[1], Vanessa Braganholo[3], Sérgio Lifschitz[1]

[1]Departamento de Informática – PUC-Rio, Rio de Janeiro, Brazil

[2]Departamento de Ciência da Computação – UERJ, Rio de Janeiro, Brazil

[3]Instituto de Computação - UFF, Niterói, Brazil

{afuentes,rogcosta,sergio}@inf.puc-rio.br
ana.almeida@ime.uerj.br, vanessa@ic.uff.br

Abstract. *Database tuning usually involves indexes, materialized views, partitioning, query rewriting and other techniques. One strategy that presents good results for performance improvements is the use of partial indexes. However, partial indexes have not been used for database tuning in the past. This is because the search space for partial indexes is exponential in the number of attributes and tuples of the table. In this paper, we address this problem by proposing an optimized strategy to select partial indexes. The optimization relies on reducing the amount of logic reads. We explain how to select the indexable attributes and their corresponding restrictions through a formal procedure. We implement our strategy to illustrate the benefits of partial indexes for tuning issues. Results are promising.*

1. Introduction

Database applications have become increasingly complex and varied. These involve very large datasets and a high demand for good performance. The problem has always been on how to decrease query response time while increasing the throughput (number of queries executed per unit of time). In this context, tuning the physical design of database systems has been proved to be extremely important in improving the performance of database systems [Shasha and Bonnet 2002].

Index tuning, as part of the physical database design, is the task of selecting, creating, deleting, and rebuilding index structures to reduce workload processing time. Among the activities related to database tuning, the adjustment of index structures represents one of the most relevant. This fact stems from the great benefit that these structures bring to the performance of database systems, since it can substantially reduce the execution time of the queries, including updates [Shasha and Bonnet 2002].

Current index tunning approaches use regular (or complete) indexes in detriment of partial index. A partial index [Stonebraker 1989], which is present in some of the major Database Management Systems, indexes a subset of the tuples of a table instead of indexing the complete set of tuples. However, the search space for defining a partial index is exponential in the number of attributes and tuples of the table, which explains why it has not been used by database tuning approaches up to now. In this paper, we propose an optimized strategy to select partial indexes. It is based in use cases oriented to reduce

the amount of logic reads. Our strategy contemplates the use of both partial and complete indexes in automatic database tuning.

It should be noted that the search for partial indexes as a tuning action includes not only all steps followed when searching for complete indexes but also the definition of the subset of tuples that will be accessed. Nevertheless, we propose a multi-column partial index approach and show that there are situations where it is worth to consider partial indexes rather than complete indexes for performance issues.

This paper is organized as follows. Section 2 defines partial indexes, while Section 3 discusses related work. In Section 4 we present a strategy to select partial indexes and combine them with complete indexes in the tuning process. Some experimental results are given in Section 5 and Section 6 concludes, listing our main contributions.

2. Preliminaries

Stonebraker defines partial indexes as constrained indexes with a WHERE clause, which defines a subset of tuples to be indexed on a table [Stonebraker 1989]. For example, we could define a partial index PI for table T with two columns A and B, as follows. In this case, only tuples from T that have attribute B valued $'X'$ or $'W'$ would be indexed.

```
CREATE INDEX PI ON T (A, B) WHERE B = 'X' or B = 'W';
```

A partial index P can be used in the execution of a query Q if and only if the predicate of Q logically implies the conditional expressions of the partial index P. For instance, consider query Q below that is run over table T:

```
SELECT * FROM T WHERE B = 'X'
```

It is easy to note that the partial index PI above contains enough information to find the tuples of T satisfying query Q, since the selection predicate of Q logically implies the selection predicate of PI. We denote this set of tuples $T(Q)$.

Partial indexes are beneficial because they can avoid indexing frequent values. When a query retrieves a set of values of more than a small percentage of all table rows, the DBMS does not use the index. Then there is no point in keeping those tuples in the index. This strategy reduces the size of the index, which speeds up the index creation time and reduces space requirements. It also speeds up many table update operations because the index does not need to be updated in all cases [PostgreSQLv9 2018].

Partial indexes may favor better performance than complete indexes because of the smaller size of their access structures. For example, consider a complete index C and a partial index PI, both using a B+-tree on attributes A and B as the physical data structure, where the partial index has a set of restrictions R. If we have a query Q for which both C and PI are useful, it might be possible that the number of scanned blocks using PI to answer Q will be smaller than the number of scanned blocks using C. Figure 1 illustrates this case for query $Q = A > 0 \; and \; A < 10 \; and \; B =' X'$. Both indexes (complete and partial) were built with a B+-tree, and the leaf nodes represent data entries that point to physical data. In this case, the number of scanned entries using the partial index can be smaller than the number of entries scanned with the complete index. For this particular example, it is in fact – while the complete index scans 21 data entries, the partial index scans only 10 data entries to answer the query Q.

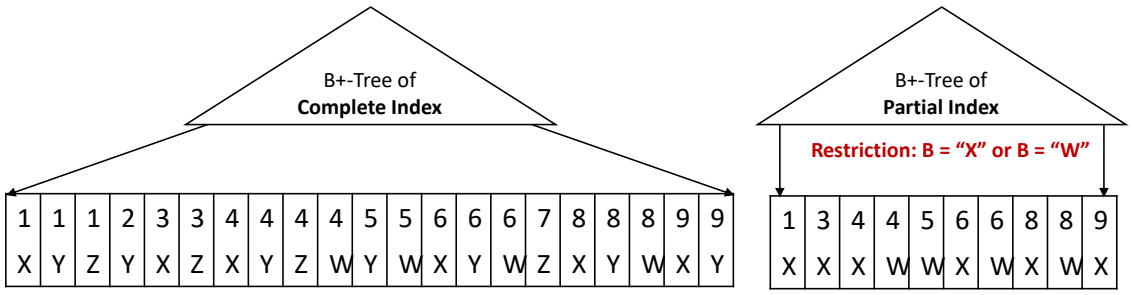


Figure 1. Representation of scanned data entries in Partial and Complete indexes for Query Q: A >0 and A<10 and B = 'X' . Values for attribute A are represented in the first row of leaf nodes, while values for attribute B are in the second row.

Finding complete indexes that may benefit a workload can be a difficult task. Besides choosing a set of attributes suitable to index creation, we also need to choose the order in which they will be indexed. There is a large number of combinations to check. The case of partial indexes is even more complex since it requires the same phases that complete indexes need and also an extra step to determine a subset of tuples which will be indexed by the partial index (the restricting selection clause).

3. Related Work

The index selection problem has been studied for many years by the database research community [Lightstone 2009]. There are two families of research in this area, which focus on development of algorithms and data structures that optimize the maintenance cost of indexes and other access structures [Labio et al. 1997], or that develop algorithms to optimize query response time [Gupta et al. 1997, Agrawal et al. 2001]. Studies about optimization of query response time may be categorized depending on how the set of candidate indexes is selected. In this work we are interest in the second family of research work [Aouiche and Darmont 2009].

Some related work about index performance focus in partial indexes as an alternative data structure. In fact, there are some cases where partial indexes are particularly useful. We discuss three situations here: (i) focus on low maintenance of the indexes; (ii) focus on saving space; and (iii) focus on reducing logical reads.

Partial indexes are useful when we have sets of tuples with a high percentage of requests and an update rate smaller than the average of updates in the table [Stonebraker 1989]. In this first situation, approaches in literature use partial indexes as a way to obtain access structures with a low maintenance cost and high performance. We may cite research work that fit to this criteria distributed in two main groups of adaptive indexing: database cracking and adaptive merging. Database cracking [Graefe and Kuno 2010a, Graefe and Kuno 2010b] combines features of automatic index and partial indexes selection by indexing results of each query. Each partitioning step creates two new sub-partitions using a logic similar to partitioning in quicksort [Idreos et al. 2011]. While database cracking functions as an incremental quicksort, with each query resulting in at most one or two partitioning steps, adaptive merging [Idreos et al. 2007a, Idreos et al. 2007b, Idreos et al. 2009, Voigt et al. 2012, Graefe et al. 2014] functions as an incremental merge-sort, with one merge step applied

to all key ranges in a query result. Under adaptive merging, the first query to use a given column in a predicate produces sorted runs. Each subsequent query on that same column applies to at most one additional merge step, that only affects those key ranges that are relevant to actual queries, leaving records in all other key ranges in their initial places [Idreos et al. 2011].

The second situation covers those cases where complete indexes are very expensive due to space constraints [Seshadri and Swami 1995]. Research work that fit in this situation [Chen et al. 2011, Wu et al. 2008] exclude from the index those sets of tuples with high selectivity, or low probability to be queried. There are tuples that do not take advantage of indexes benefits and lack of interest in the current workload.

Last but not least, the third situation is the one we explore in this paper. We claim that the use of multi-column partial indexes is a good alternative to improve query performance. We remove the space constrains of the second alternative and introduce a new view of partial indexes to reduce the number of logical reads in a workload.

4. Tuning with partial indexes

We advocate that the tuning process must take into consideration solutions containing both partial and complete indexes. The selection process of partial indexes can be divided into the following phases: (i) selection of indexable attributes (ii) definition of restrictions for partial indexes and (iii) final index configuration. These steps should ensure that any partial index chosen to be part of the final access structure configuration of a database system has a high probability of being used. Moreover, it should bring actual benefits during the workload execution. Each of these phases is discussed next.

4.1. Step 1: Selection of indexable attributes

An index is a copy of values from selected columns of a table that can be searched very efficiently. It includes a low-level disk block address or a direct link to the complete row of data it was copied from. The order in which the index definition specifies the columns is important. It is possible to retrieve a set of row identifiers using only the first indexed column. However, it is not possible or efficient (on most databases) to retrieve the set of row identifiers using only the second or greater indexed column(s).

We call indexable attributes of a query the sets of attributes belonging to the same table operated by the AND operator for which it makes sense to create an index. For example, in the following query, the set of possible sets of indexable attributes is A_t = {{A},{B},{A,B}}.

```
SELECT * FROM T WHERE A > 2 and  B = 'X'
```

In order to obtain the sets of indexable attributes in a workload, we perform an automatic analysis of the predicate at the WHERE clause of each query. The analysis derives new clauses expressed as a disjunction of conjunctions, that is equivalent to the original clause. The procedure consists in modeling the operations in the WHERE clause in a tree representing the priority of operations and rotating the tree according to the equivalence of operations until no node representing an OR logic operation has an AND node as its parent. Figure 2(a) represents a clause (A OR B) AND C, where internal nodes are logic operations and leaf nodes are restrictions (predicates). Figure 2(b) represents a rotation

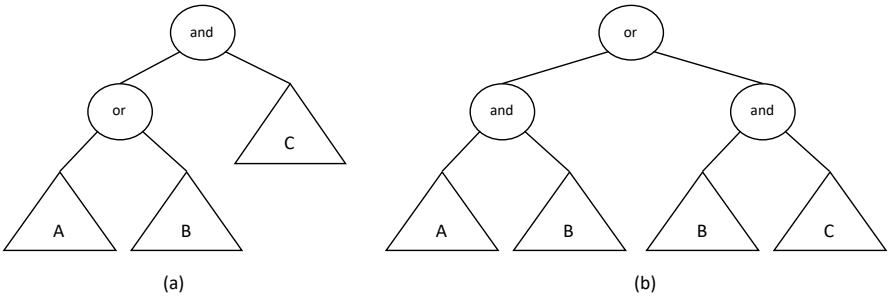


Figure 2. (a) Tree of operations of a given query; (b) Rotation of the tree in (a) so that no OR operation has an AND operation as its parent

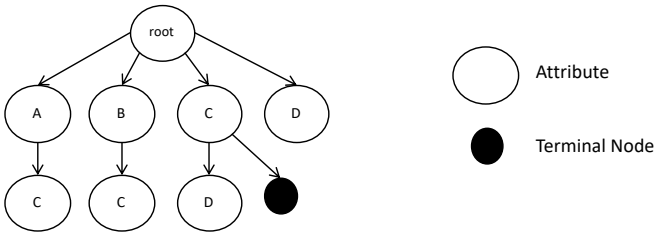


Figure 3. Trie representing a conceptual lattice

of Figure 2(a) once a node OR with a parent AND is found. We use this modified tree to obtain the indexable attributes, which in this case are $\{\{A\}, \{B\}, \{C\}, \{A, B\}, \{B, C\}\}$.

Since indexes are defined over attributes of the same table, the second step resides in separating attributes that belong to the same table. The sets of indexable attributes are obtained by looking for sets of restrictions (predicates) with attributes in the same table belonging to the same subtree with AND root node type. Once the sets of indexable attributes for each query are obtained, we may find some sets of indexable attributes repeated multiple times. These sets are those that appear more frequently in the queries of the workload, therefore, they are candidates for the creation of partial indexes.

We use a method based on conceptual lattices [Gouda and Zaki 2001] (hierarchical structure under the subset operation representing a partial order) to obtain frequent sets of indexable attributes. The data structure used for the representation of the conceptual lattices in this work is a Trie [Fredkin 1960], that counts the number of times that each set of attributes appear in the workload. Figure 3 shows a Trie representing the sets of indexable attributes $A_t = \{\{A, C\}, \{B, C\}, \{C, D\}, \{D\}\}$. In this example, if a set having three or more occurrences is considered frequent, then the attribute C must be marked as frequent and a terminal node is associated.

A set of indexable attributes is considered frequent if the proportion between the set of indexable attributes and the total number of extracted sets of indexable attributes is above a given threshold t, assuming an uniform distribution. This gives us an estimate of the number of times a given index would be useful in a query workload. The threshold t is calculated as $1/|A_t|$, where $|A_t|$ is the cardinality of the set of indexable attributes.

Once the sets of indexable attributes of each table are determined, we need a strategy to assess the impact of the candidate indexes in the workload performance. In other

words, we need to identify the indexes that could be defined over the sets of indexable attributes. Hence, we adopt an heuristics of benefits to evaluate candidate indexes over sets of indexable attributes. This heuristics consists of finding, for each query of the workload, a set of candidate indexes with the higher benefit to the query according to a cost model. To do so, we obtain an aggregation of gains by estimating, for each query, the cost of running that query without the index minus the cost of executing the query with the index. Then, we obtain the benefit of that particular index by calculating the aggregated gain of the index minus its maintenance cost.

4.2. Step 2: Selection of restrictions

Let Tab be a table having a set of attributes A and a set of tuples T, where the round number of tuples fitting into a block is d. Let CI be a complete index defined over a set of indexable attributes A_t and PI a partial index defined over the same set of indexable attributes A_t having the set of restrictions RS. Each element $R \in RS$ represents a set of restrictions (predicates) relative to a subset of indexable attributes of A_t. For example, consider the definition of the partial index PI below. We can represent this partial index by a set of indexable attributes $A_t = \{A, B\}$ and the set of predicates $RS = \{\{B = 'X'\}, \{B = 'W'\}\}$.

```
CREATE INDEX PI ON T (A, B) WHERE B='X' or B='W';
```

Moreover, let Q be a query represented by a set of indexable attributes $A_q = \{a_1, a_2, ..., a_n\}$ and a set of predicates $Q_p = \{p_1, p_2, ..., p_n\}$, where each predicate p_i restricts a subset of attributes of A_q. For example, considering a table Tab with the attributes $\{A, B, C\}$ the query defined below can be represented by the sets $A_q = \{A, B\}$ and $Q_p = \{\{A > 2, B =' X'\}\}$.

```
SELECT * FROM T WHERE A > 2 and B = 'X'
```

To calculate the profit (gain) G of a partial index, we find the difference between the number of blocks scanned using the complete index CI and the number of blocks scanned using the partial index PI. Equation 1 shows a formula for estimating G by calculating the proportion of non scanned tuples to answer query Q and multiplying it by the total number of blocks in the table. In the formula, $T(C)$ denotes the number of tuples satisfying C. C can be either a query or the restrictions on an index. T is the total number of tuples in the table. For each $a_i \in A_q$, we define $CQ_i = T(X)$ and $CQ'_i = T(Y)$, such that X and Y are the set of predicates (restrictions) p_i ($X \subseteq p_i$, $Y \subseteq p_i$ and $X \neq Y$) associated to the set of indexable attributes of the set $A_t \cap a_i$ and $a_i \setminus A_t$ respectively.

$$G = \frac{(1 - \frac{T(PI)}{T}) * \frac{CQ_i}{T} * T}{d} \tag{1}$$

We then define the amount of scanned blocks B for executing the query Q (Equation 2). The notation $SB(CQ_i, CQ'_i)$ represents the amount of blocks scanned in the table when using an index.

$$B = \frac{CQ_i}{d} + SB(CQ_i, CQ'_i) \tag{2}$$

We assume in Equation 1 that the non-scanned tuples have a uniform distribution over the key A_t of the index. Moreover, the term $SB(CQ_i, CQ'_i)$ can be calculated using the results in [Mackert and Lohman 1989], that estimate the number of disk page fetches when randomly accessing k records out of n given records stored on p disk pages.

When we divide Equation 1 by Equation 2, the result quantifies how good is the benefit of using the partial index compared to the execution of Q using the complete index. Equation 3 assumes we are only interested in partial indexes having a benefit greater than a threshold w.

$$w < \frac{(1 - \frac{T(PI)}{T}) * \frac{CQ_i}{T} * T}{\frac{\frac{CQ_i}{d} + SB(CQ_i, CQ'_i)}{d}} \tag{3}$$

Manipulating Equation 3, we have that:

$$\frac{T(PI)}{T} < 1 - w * (1 + \frac{SB(CQ_i, CQ'_i) * d}{CQ_i}) \tag{4}$$

The partial index would be used only if the cost is less than a full scan. Let TB be the number of blocks occupied by the tuples of table Tab, assuming an uniform distribution:

$$\frac{T(PI)}{T} * \frac{CQ_i}{d} + SB(CQ_i, CQ'_i) < TB \tag{5}$$

Then, from Equations 4 and 5 it is possible obtain:

$$\frac{T(PI)}{T} < min(1 - w * (1 + \frac{SB(CQ_i, CQ'_i) * d}{CQ_i}), \frac{TB - SB(CQ_i, CQ'_i)}{\frac{CQ_i}{d}}) \tag{6}$$

Each restriction belonging to PI must comply to Equation 4 ensuring the partial index does reduce the number of logical reads. However, we also need to ensure that tuples in a partial index will have a high probability of being accessed. We define a similarity function $s(p_1, p_2)$ between predicates p_1 and p_2 and a ranking function *rank*(p, Q_p) for predicate p regarding the set of predicates Q_p as:

$$s(p_1, p_2) = \frac{|T(p_1) \cap T(p_2)|}{|T(p_1) \cup T(p_2)|}, \tag{7}$$

$$rank(p, Q_p) = \sum\nolimits_{p_i \in Q_p} s(p, p_i) \tag{8}$$

Then, it is possible define the restrictions of a partial index PI defined in a set of attributes A_t using the set of restrictions RS belonging to any set of indexable attributes A

in the queries of a workload, just by solving the following integer programming problem. In this program, variable a_i must have an integer value in the set $\{0\ ,\ 1\}$. This value denotes if the restriction r_i can be part of the restrictions of PI.

$$\text{maximize} \quad \sum_{r_i \in RS} a_i * rank(r_i, RS), \text{subject to Equation 6}$$

4.3. Step 3: Final configuration

We have so far presented a methodology that finds indexable attribute sets for a given workload. These sets are candidates for the creation of either partial or complete indexes. Furthermore, let C be an indexable attribute set, w a workload and RS a set of conjunctions belonging to queries in w involving constraints on attributes in C. We present a method that may obtain a set P of conjunctions that together with those constraints in C represent a partial index noted by (C, P). The corresponding complete index (C, U) takes into account the universal set of all possible conjunctions.

However, the whole process may generate multiple partial and complete candidate indexes, as we do not consider the relationship of these indexes with the workload. We may cite as an example a table T containing attributes A, B, C and D, and query Q_1 in SQL in workload w:

```
Q1: SELECT * FROM T
    WHERE A > 1 and B > 2 and C > 3 and D > 4
```

Now, consider six possible indexable attribute sets that are frequent in w: $\{\{A\}, \{B\}, \{C\}, \{D\}, \{A, B\}, \{C, D\}\}$. These sets imply complete candidate indexes that could be used by query Q_1 separately. Thus, some of them would become redundant for this query. To avoid such a situation, we consider an algorithm that outputs all possible distinct candidate indexes that may be obtained from all candidate indexes generated in previous steps.

Let C_i be a set of indexable attributes that are frequent in a workload w and P_i a set of constraints. Function f_O computes the execution cost considering w and candidate structures in O as input. Note that the i^{th} candidate structure would be a complete index if $P_i = U$.

$$O_c = \bigcup_{i=1}^{k} < C_i, P_i >$$

Therefore, it is possible to run an algorithm that starts with an empty candidate structures set O and at each step, adds candidate structure C in O_c that together with those structures already in O can minimize the execution cost for w concerning f_O. That is, at each step we determine C in O_c that minimizes $f_O(\{< C_i, P_i >\} \cup O)$. Once available, C will be added to O whenever it increases the minimum value for f_O in the previous step.

5. Experiments

Our work intends to show that, in some cases, partial indexes may be used to improve performance rather than complete indexes. This happens because multicolumn partial indexes, with any non-categorical column in the left-most column and restrictions in any of the right-most column, can reduce the amount of logic reads in queries using the partial index when compared to the corresponding complete index.

In order to show that partial indexes may reduce the amount of logical reads, we implemented a prototype of the proposed solution in the DBX [BiobdPUC-Rio 2018] framework, whose architecture enables incorporating several database tuning solutions. Our prototype is used to generate configurations consisting of both partial and complete indexes, and also configurations consisting of only complete indexes as a result of the workload execution.

Methodology. All our experiments were run on PostgreSQL v9 DBMS, on a 64 bit computer with a 1.6GHz Quad-Core processor, 16GB of RAM and 1TB hard disk drive. We have used a database obtained from the TPC-H benchmark, configured to a scale of 100 GB. Just to mention a few examples, in this case the largest table (LineItem) has over 600 million tuples and 90MB. The supplier table has a smaller number of tuples (30,000) but uses more than 170MB, almost twice as much as LineItem.

We instantiated 8 queries out of 22 generic queries in the TPC-H benchmark. These involve simple queries in the sense that no nested queries are taken into account. We have generated 100 queries using these TPC-H queries by randomly choosing a generic query at each time. The execution of the experiment was developed in two phases. The first phase consisted of the execution of the workload using DBX to generate configurations of complete indexes only. In the second phase, DBX was tailored to generate configurations of both partial and complete indexes. This way, it was possible to determine how good partial indexes might be when compared to complete indexes. We have chosen to compare the behavior of the query execution times and also the number of logical reads for each query. It is worth noting that the time spent in query execution was measured by executing each query 10 times, discarding the first measure and computing the average of the remaining executions.

Results and discussion. The first thing we could find out about partial indexes in our research is that for single column indexes, there is no difference in the performance of query execution between partial indexes and complete indexes. This is reflected in the Equation 3 where the number of non scanned tuples for the partial index is 0.

Using the configurations suggested by the prototype implemented in DBX, we have compared the configurations of partial indexes and complete indexes, with the corresponding configuration of only complete indexes, by measuring the amount of logic reads and the execution times. Figure 4 shows the number of logic reads. The difference of logic reads in the generic queries $Q4$, $Q6$ and $Q8$ is due to the existing partial indexes. We get a reduction of logical reads of approximately 8%, 47%, and 64%, which shows that our strategy to select partial indexes have achieved good results. Note that $Q4$, $Q6$ and $Q8$ are the only queries in the workload for which partial indexes were created. This is way the amount of logic reads for the other queries are the same.

When we analyze the query execution plan for $Q8$, we note that the complete

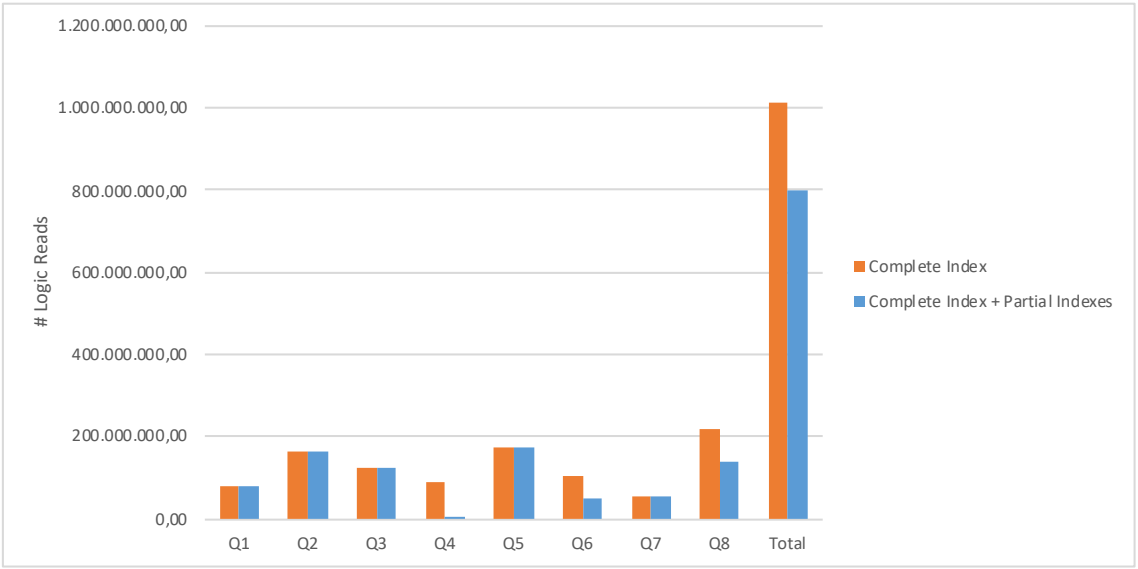


Figure 4. Logic reads by each generic query

Table 1. Hypothesis test for average difference

Query	#Instances	Complete Index + Partial Index		Complete Index		p-value
		AVG	σ	AVG	σ	
Q4	14	784.920	6.282	1.680.707	8.323	0.0001
Q6	7	1.788.024	10.371	2.595.876	13.625	0.0001
Q8	22	896.590	9.215	1.172.261	9.578	0.0001

index is not used. This happens because the partial index reduces the number of blocks to be read when compared to the complete index. A reduction in the amount of logic reads implies a reduction in the query execution time. Indeed, Figure 5 shows that the execution time for queries $Q4$, $Q6$ and $Q8$ do decrease.

The next step of our evaluation comprises checking whether the reduction on query processing time when using partial indexes for queries $Q4$, $Q6$ and $Q8$ are statistically significant. To show this, we have considered the hypothesis test of mean (or average) difference. This test is based on an attempt to reject the null hypothesis that two variables are equal. Specifically, we intend to reject the hypothesis that the execution time for queries $Q4$, $Q6$ and $Q8$ is the same for the configuration with both complete and partial indexes and the configuration with only complete indexes. Table 1 shows the results of our hypothesis test. In this table, *#Instances* refers to the number of instances of the generic query we used (please note that each instance was executed 10 times, as mentioned before). AVG is the mean of query execution time, and σ is the standard deviation. For those three queries, the difference in the mean execution time is statistically significant ($p\text{-}value < 0.05$).

6. Conclusions

This work shows that partial indexes are useful data access structures capable of improving query execution time. Particularly, we show that multi-column partial indexes can be

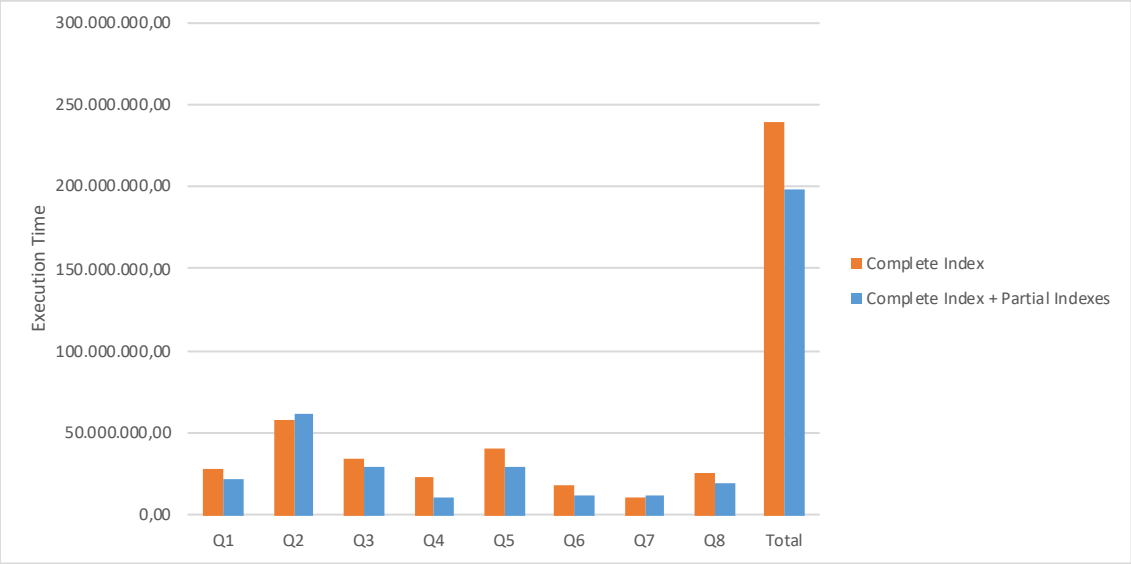


Figure 5. Execution time by each generic query

used to improve performance since it can reduce the amount of logical reads needed to process a query. We have identified at least two situations where multi-column partial indexes are effective: (i) for sets of tuples frequently queried and (ii) when there are very selective attributes. These are very common situations in practice. Therefore, it is worth counting on partial index as an additional way of obtaining efficient database systems.

Our contribution resides in an approach for database tuning that is capable of generating configurations for both partial and complete indexes. The proposed strategy is based on use cases identified for partial indexes. We have implemented and tested our proposal considering a standard OLTP benchmark (TPC-H). The obtained results are promising and show situations where partial indexes are used by the query optimizer even when the corresponding complete indexes are present. We invite the reader to check other results at [Fuentes 2016].

We intend to continue this research work by further studying the influence of the order of attributes in partial indexes. In addition, our approach could also be considered by automatic database tuning tools.

References

Agrawal, S., Chaudhuri, S., and Narasayya, V. R. (2001). Materialized view and index selection tool for microsoft SQL server 2000. In *SIGMOD Conf.*, page 608. ACM.

Aouiche, K. and Darmont, J. (2009). Data mining-based materialized view and index selection in data warehouses. *J. Intell. Inf. Syst.*, 33(1):65–93.

BiobdPUC-Rio (2018). Dbx. `https://github.com/BioBD/dbx`. [April-03-18].

Chen, C., Li, F., Ooi, B. C., and Wu, S. (2011). TI: an efficient indexing mechanism for real-time search on tweets. In *SIGMOD Conference*, pages 649–660. ACM.

Fredkin, E. (1960). Trie memory. *Commun. ACM*, 3(9):490–499.

Fuentes, A. D. (2016). Automatic fine tuning with partial indexes (in portuguese). Master's thesis, PUC-Rio Informática, Rio de Janeiro, Brasil.

Gouda, K. and Zaki, M. J. (2001). Efficiently mining maximal frequent itemsets. In *ICDM*, pages 163–170. IEEE Computer Society.

Graefe, G., Halim, F., Idreos, S., Kuno, H. A., Manegold, S., and Seeger, B. (2014). Transactional support for adaptive indexing. *VLDB J.*, 23(2):303–328.

Graefe, G. and Kuno, H. A. (2010a). Adaptive indexing for relational keys. In *ICDE Workshops*, pages 69–74. IEEE Computer Society.

Graefe, G. and Kuno, H. A. (2010b). Self-selecting, self-tuning, incrementally optimized indexes. In *EDBT*, Intl Conf, pages 371–381. ACM.

Gupta, H., Harinarayan, V., Rajaraman, A., and Ullman, J. D. (1997). Index selection for OLAP. In *ICDE*, pages 208–219. IEEE Computer Society.

Idreos, S., Kersten, M. L., and Manegold, S. (2007a). Database cracking. In *CIDR*, pages 68–78. www.cidrdb.org.

Idreos, S., Kersten, M. L., and Manegold, S. (2007b). Updating a cracked database. In *SIGMOD Conference*, pages 413–424. ACM.

Idreos, S., Kersten, M. L., and Manegold, S. (2009). Self-organizing tuple reconstruction in column-stores. In *SIGMOD Conference*, pages 297–308. ACM.

Idreos, S., Manegold, S., Kuno, H. A., and Graefe, G. (2011). Merging what's cracked, cracking what's merged: Adaptive indexing in main-memory column-stores. *PVLDB*, 4(9):585–597.

Labio, W., Quass, D., and Adelberg, B. (1997). Physical database design for data warehouses. In *ICDE*, pages 277–288. IEEE Computer Society.

Lightstone, S. (2009). Physical database design for relational databases. In *Encyclopedia of Database Systems*, pages 2108–2114. Springer US.

Mackert, L. F. and Lohman, G. M. (1989). Index scans using a finite LRU buffer: A validated I/O model. *ACM Trans. Database Syst.*, 14(3):401–424.

PostgreSQLv9 (2018). Partial indexes documentation. `http://www.postgresql.org/docs/9.4/static/indexes-partial.html`. [April-03-18].

Seshadri, P. and Swami, A. N. (1995). Generalized partial indexes. In *ICDE*, pages 420–427. IEEE Computer Society.

Shasha, D. E. and Bonnet, P. (2002). *Database Tuning - Principles, Experiments, and Troubleshooting Techniques*. Elsevier.

Stonebraker, M. (1989). The case for partial indexes. *SIGMOD Record*, 18(4):4–11.

Voigt, H., Jäkel, T., Kissinger, T., and Lehner, W. (2012). Adaptive index buffer. In *ICDE Workshops*, pages 308–314. IEEE Computer Society.

Wu, S., Li, J., Ooi, B. C., and Tan, K. (2008). Just-in-time query retrieval over partially indexed data on structured P2P overlays. In *SIGMOD Conf.*, pages 279–290. ACM.

FLEXMVCC: Uma abordagem flexível para protocolos de controle de concorrência multi-versão

Eder C. M. Gomes[1], **J. Filipe L. de Sousa**[1],
Paulo R. P. Amora[1], **Javam C. Machado**[1]

[1]Laboratório de Sistemas e Bancos de Dados (LSBD)
DC/UFC – CEP 60440-900 – Fortaleza – CE – Brazil

{eder.clayton,filipe.lobo,paulo.amora,javam.machado}@lsbd.ufc.br

Abstract. *Different concurrency control protocols impact the performance of database systems depending on the workload profile. Without prior knowledge of this workload and its changes, the decision on which protocol to use becomes challenging. To alleviate this impact, we created FLEXMVCC, which integrates two compatible multi-version concurrency control protocols, an optimistic one and a pessimistic one, able to adapt as the workload changes. Preliminary experiments show that the exchange between protocols is feasible and results in a performance gain over static protocols.*

Resumo. *Diferentes protocolos de controle de concorrência impactam o desempenho do banco de dados, dependendo do perfil da carga de trabalho. Sem conhecimento prévio desta carga e suas mudanças, a decisão na escolha de um protocolo se torna desafiadora. Para tanto, criamos o FLEXMVCC, que integra dois protocolos de controle de concorrência multi-versão compatíveis, um otimista e um pessimista, capaz de se adaptar conforme as mudanças na carga de trabalho. Experimentos preliminares mostram que a troca entre protocolos é viável e representa um ganho de desempenho em relação aos protocolos estáticos.*

1. Introdução

Com o aumento do número de núcleos nos processadores atuais e a queda de preço das memórias RAM no mercado é cada vez mais viável um banco de dados ser armazenado inteiramente em memória e consequentemente o controle de concorrência em um sistema de banco de dados se torna um gargalo por que o processamento de consulta em memória é tão rápido quanto a execução do protocolo [Harizopoulos et al. 2008, Yu 2015].

Existem diversos controles de concorrência para inúmeras situações de acesso ao dado, por exemplo, cargas de trabalho OLTP e OLAP, sendo que cada controle de concorrência vai ter um melhor desempenho para um conjunto dessas situações e infelizmente não há um controle de concorrência que tenha o melhor desempenho para todas as situações de acesso ao dado. Cabe então ao administrador do banco de dados escolher, de acordo com o acesso aos dados, a melhor alternativa dentre os controles de concorrência disponíveis. Mas nem sempre o acesso aos dados segue um padrão único, podendo diversificar ao decorrer do tempo do banco de dados.

Tenhamos como exemplo um supermercado, no qual corriqueiramente há atualizações em seus dados (por exemplo venda de mercadoria) e leitura dos dados (por exemplo

verificação de estoque) e o administrador do banco de dados escolhe com base nessa carga de trabalho padronizada a melhor configuração do banco de dados. Sabemos que o mundo é dinâmico, há mudanças no ambiente que geram novos acessos aos dados, por exemplo, o evento *Black Friday* que aumenta as atualizações nos dados e auditoria na empresa que aumenta as leituras nos dados para geração de relatórios e em nenhuma dessas situações o banco de dados pode ter um desempenho abaixo do esperado, mas como diverge do acesso padrão estabelecido pelo administrador do banco de dados, acaba tendo perda de desempenho.

Pensando nisso, criamos um gerenciador de transações para controle de concorrência multi-versão (MVCC) chamado FLEXMVCC que permite uma execução flexível e correta entre os protocolos otimista para multi-versão (MVOCC) [Larson et al. 2011] e *two-phase locking* (MV2PL) [Larson et al. 2011]. Também desenvolvemos um método geral de reconfiguração de protocolo para suportar as mudanças online entre os protocolos. A essência desta abordagem é que no período de transição de um protocolo ao outro é criado um protocolo compatível com os protocolos antigo e novo, de modo que o processo de troca não precise interromper todos as transações, ao mesmo tempo em que minimiza o impacto na taxa de transferência e latência.

2. Trabalhos Relacionados

Alguns projetos propõem novos protocolos de controles de concorrência otimizados para bancos de dados de memória principal, como Doppel [Narula et al. 2014] propõe um protocolo escalável de controle de concorrência para operações comutativas sob cargas de trabalho de alta contenção. Tictoc [Yu et al. 2016] extrai mais concorrência ao avaliar os *timestamps* das transações. Hekaton [Diaconu et al. 2013, Larson et al. 2011] melhora o desempenho dos protocolos 2PL e OCC usando tabelas *hash* sem múltiplas versões para bancos de dados de memória principal. Silo [Tu et al. 2013] desenvolve um protocolo OCC que atinge alto desempenho evitando todos os pontos de contenção centralizados entre núcleos de CPU. Nesses trabalhos, o foco principal é no desempenho quando há um hardware com um grande número de núcleos de processamento e novos hardwares com uma alta taxa de I/O, mas não se atentam em ter uma boa performance em situações mistas.

Outros trabalhos exploram a combinação de protocolos de controle de concorrência em um sistema de banco de dados, como no trabalho [Xie et al. 2015] e na continuação do seu trabalho [Su et al. 2017] em que é fornecido uma modularização apresentando um protocolo de controle de concorrência para cada grupo com base em uma análise da carga de trabalho off-line. Para resolução de conflitos transacionais entre grupos é apresentado um protocolo base para resolve-lo. Embora a atribuição de protocolo orientada ao procedimento armazenado possa processar conflitos dentro do mesmo grupo de forma mais eficiente, a sobrecarga de controle de concorrência adicional de executar protocolos entre grupos pode se tornar um gargalo de desempenho para um banco de dados de memória principal. Além disso, o agrupamento baseado na análise off-line assume que os conflitos de carga de trabalho são conhecidos por conta própria, o que pode não ser verdade em aplicativos reais.

Um outro trabalho que mistura protocolos é o [Wang and Kimura 2016]. Esse trabalho consiste na criação do protocolo *Mostly-optimistic concurrency control* (MOCC)

com base no OCC e com a adição de bloqueios similares ao 2PL, mas com modificações capazes de melhorar o desempenho do protocolo para equipamentos com milhares de núcleos, mas para cargas de trabalho de baixo conflito, o MOCC mantém o desempenho padrão do protocolo OCC não se preocupando na diversidade da carga de trabalho abordada no nosso trabalho. O [Shang et al. 2016] também mistura os protocolos OCC e 2PL, mas se restringe a melhorar o desempenho de conjuntos de dados gráficos.

3. MVCC - Controle de concorrência multi-versão

De acordo com [Yu 2015] o esquema mais popular usado nos SGBDs desenvolvidos na última década é o controle de concorrência multi-versão (MVCC). A principal diferença desse esquema para esquemas com uma versão é que o SGBD mantém várias versões físicas de cada objeto lógico no banco de dados para permitir que as operações no mesmo objeto continuem em paralelo, dessa forma, as transações somente leitura acessam versões mais antigas de tuplas, sem impedir que as transações de leitura e gravação gerem versões mais novas simultaneamente. [Yu 2015] continua expondo que para se obter as multi-versões é necessário que para cada escrita em uma tupla gere uma nova versão daquela tupla.

Independente do protocolo implementado, há metadados comuns, tanto para tuplas como para transações. Nas transações, é atribuído um *timestamp*, como identificador único (Tid) no SGBD, quando a transação é iniciada no sistema. Os protocolos de controle de concorrência usam esse identificador para marcar quais versões de tupla essa transação acessa. As tuplas são compostas por uma ou mais versões e em cada versão há 4 campos de metadados além do conteúdo da tupla que as diferencia.

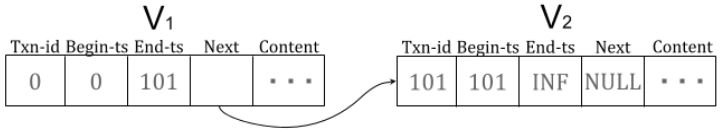


Figura 1. Exemplo de versões de uma tupla e seus metadados

O primeiro campo é o *txn-id* que tem como objetivo ser o bloqueio de escrita para a versão. As versões tem o valor zero nesse campo quando não há bloqueios de escrita. Se uma transação com identificador T_{id} quiser atualizar uma versão de uma tupla X, o SGBD verifica se o campo *txn-id* é zero, caso positivo o SGBD irá definir o valor de *txn-id* para T_{id}. Na Figura 1 temos um exemplo de duas versões, a V1 sem bloqueio e a V2 com bloqueio de escrita. Os próximos dois campos são os *timestamps* de começo e final que representam o tempo de vida da versão da tupla. O SGBD define quais tuplas são visíveis a uma transação com identificador T_{id} verificando se o T_{id} está entre esses campos. As tuplas são inicializadas com zero no *begin-ts* e *INF* no *end-ts*. Na Figura 1 temos um exemplo de duas versões, a V1 visível para transações com *timestamp* entre 0 e 100 e V2 visível para transações a partir de 101. O último campo chamado de *Next* armazena o endereço para a próxima versão da tupla, formando assim uma lista encadeada de versões.

3.1. MVOCC - Otimista

O MVOCC [Larson et al. 2011] é um protocolo otimista para multi-versão baseado no protocolo otimista para uma versão proposto em 1981 [Kung and Robinson 1981]. A

motivação por trás do protocolo otimista é que assume-se que a probabilidade de que as transações conflitem é miníma, dessa forma as transações são liberadas para efetuar suas operações, sem se importar em adquirir bloqueios de leitura e escrita e na etapa de confirmação é verificado se as transações podem confirmar suas operações, caso elas não possam, a transação é abortada.

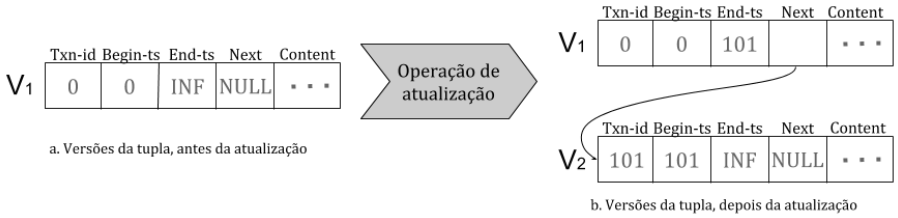


Figura 2. Operação de atualização em um tupla usando o protocolo MVOCC

Esse protocolo não precisa de adição de metadados nas e divide uma transação em três fases, a execução, a validação e a confirmação. Na fase de execução é onde acontece as operações de leitura e escrita da transação. A operação de leitura consiste em procurar a versão visível mais recente com base nos campos *begin-ts* e *end-ts*, e efetua a leitura do conteúdo dessa tupla. A operação de escrita consiste em procurar a versão visível mais recente e criar uma nova versão com base nela, mas só é permitido a escrita caso o *Txn-id* dessa versão seja igual a zero. Quando é necessário confirmar, entra na fase de validação e é atribuído a um novo *timestamp* chamado T_{commit} para determinar a ordem das transações. O SGBD então determina se as tuplas no conjunto de leitura da transação foram atualizadas por uma transação já confirmada, caso positivo a transação é abortada. Passando por essas verificações, a transação entra na fase de gravação na qual o SGBD instala todas as novas versões e define os campos *begin-ts* para T_{commit} e *end-ts* para *INF*.

3.2. MV2PL - Bloqueio em duas fases

O MV2PL [Larson et al. 2011] é um protocolo pessimista para multi-versão baseado no protocolo *two-phase lock* para uma versão [Bernstein et al. 1987]. Diferente do protocolo MVOCC, a motivação para esse protocolo é que assume-se que a probabilidade de que as transações conflitem é alta, dessa forma para se efetuar uma operação é necessário adquirir o bloqueio pela versão requisitada. Nesse protocolo existem dois tipos de bloqueios, o bloqueio de escrita e o bloqueio de leitura compartilhada. Já existe um campo nos metadados já explicados para o bloqueio de escrita, o campo *txn-id*, mas para leitura não há, portanto é necessário criar um campo extra, com nome de *read-cnt*, como visto na Figura 3. Esse novo campo serve para ser um contador de leituras para uma versão especifica.

Esse protocolo divide a transação em duas fases, a fase de expansão e a fase de recolhimento. Na fase de expansão é onde acontece a obtenção dos bloqueios de leitura e escrita e na fase de recolhimento acontece a liberação dos bloqueios e a confirmação das operações feitas pela transação. Na operação de leitura, o SGBD busca a versão da tupla visível mais recente e verifica se não há bloqueios de escrita verificando se o campo *txn-id* é igual a zero, depois é incrementado o valor do campo *read-cnt*, assim como está representado na transição das Figuras 3a para 3c. Na operação de escrita, o SGBD busca

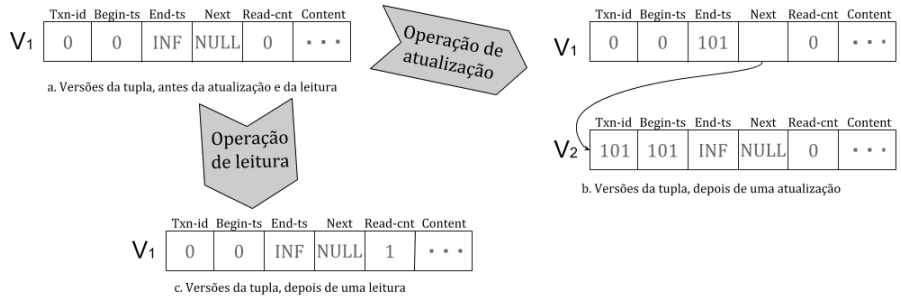


Figura 3. Operação de atualização e leitura em um tupla usando o protocolo MV2PL

a versão da tupla visível mais recente e verifica se há bloqueios de escrita ou leitura, verificando se os campos *txn-id* e *read-cnt* estão com valor zero, se estiverem livres de bloqueio é criado uma nova versão atualizada, assim como está representado na transição das Figuras 3a para 3b. Na confirmação da transação, é gerado um T_{commit} que é usado para atualizar o campo *begin-ts* para as versões criadas por essa transação e, em seguida, libera todos os bloqueios da transação.

4. Arquitetura para gerenciadores de transações

Quando se tem mais de um protocolo de controle de concorrência para ser gerenciado, a arquitetura padrão de um gerenciador de transações não é mais satisfatório, ou seja, todas as transações usarem apenas um protocolo para acessar todos os registros não é mais o suficiente. Em [Tanger 2017] são criadas 4 alternativas de arquitetura para SGBDs que usam mais de um controle de concorrência, que podem ser visualizados na Figura 4. Discutiremos agora as desvantagens com relação ao nosso trabalho dentre as possibilidades expostas.

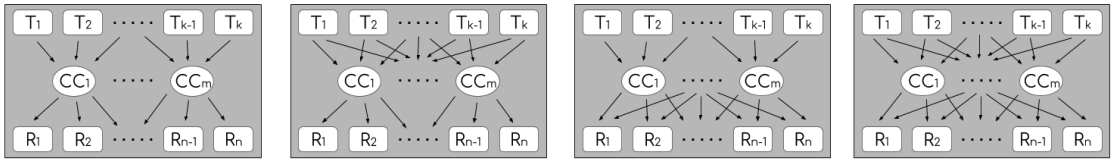


Figura 4. Arquiteturas para controle de concorrência misto. Fonte (adaptado): [Tanger 2017]

A primeira arquitetura é a mais simples e pode ser encontrado na Figura 4a. Abordamos aqui uma arquitetura em que cada transação (denotado por T) só pode escolher um protocolo de controle de concorrência (denotado por CC) e cada registro (denotado por R) pode ser acessado através de apenas um protocolo. Na Figura 4b temos uma outra arquitetura que consiste em qualquer transação pode escolher um ou mais protocolos ao decorrer de sua existência e cada registro pode ser acessado apenas por um protocolo. Continuando na Figura 4c, cada transação pode escolher apenas um protocolo de controle de concorrência e os registros podem ser acessados por qualquer controle de concorrência. A última arquitetura, encontrado na Figura 4d, é a abordagem que temos que qualquer transação

pode escolher um ou mais protocolos ao decorrer de sua existência e cada protocolo pode acessar qualquer registro.

Nas arquiteturas das Figuras 4a e 4b precisamos de um estudo sobre os dados antes da execução do banco de dados, pois para usarmos essas arquiteturas, com tuplas sendo acessadas por apenas um protocolo, seria necessário manter dados separados de acordo com o protocolo de controle de concorrência e para se atingir um bom desempenho, cada protocolo deve ser o melhor para o seu conjunto de dados. Como o nosso trabalho pretende não ter a necessidade de se ter um estudo prévio dos dados, essas duas arquiteturas foram descartadas.

Nas arquiteturas das Figuras 4b e 4d, temos que as transações podem escolher um ou mais protocolos ao decorrer de suas existências. Essa troca de protocolos abre possibilidades de tratamento para um maior número de casos específicos durante a troca de protocolos em uma mesma transação, aumentando assim a sobrecarga no banco. Portanto, essa sobrecarga pode se tornar grande o suficiente e consequentemente penalizando nossa estratégia.

Por fim, ficamos satisfeitos com a arquitetura da Figura 4c, pois os dados estão disponíveis para qualquer protocolo de controle de concorrência e as transações iniciam e finalizam sem mudanças de protocolo. Com alguns ajustes no gerenciamento dos metadados existentes em protocolos MVCC, podemos conseguir com que duas ou mais transações usando protocolos diferentes possam ser gerenciadas pelo SGBD.

5. FLEXMVCC

O FLEXMVCC tem como proposta ser um controle de concorrência multi-versão flexível a mudanças entre os protocolos MVOCC e MV2PL.

O principal objetivo da nossa proposta é que quando a carga de trabalho for melhor para o protocolo MVOCC o FLEXMVCC escolha-o como protocolo ativo no SGBD, e quando a carga de trabalho for melhor para o protocolo MV2PL o FLEXMVCC escolha-o. Com isso a tendência é que o banco de dados ganhe desempenho em cargas de trabalho mista, ganhando as vantagens que cada protocolo pode oferecer para cargas de trabalho especificas.

Para que não haja perda no desempenho entre a troca de protocolos, ou seja, para que o SGBD não espere todas as transações de um protocolo finalizarem para começar as transações do outro protocolo iniciarem, foi necessário criar um período de transição entre a troca de protocolo, para que os dois protocolos convivam no SGBD ao mesmo tempo até que todas as transações que estão executando no protocolo antigo finalizem.

Estudando os protocolos, percebemos que o bloqueio de escrita do MV2PL já está presente no MVOCC, por meio do campo *txn-id*, mas o bloqueio de leitura não é implementado no MVOCC. Levando isso em consideração adicionamos o campo *read-cnt* no protocolo MVOCC e pequenas alterações na operação de leitura e confirmação do protocolo MVOCC.

Na operação de leitura, como no MV2PL, adicionamos o incremento no campo *read-cnt* com intuito de que o SGBD saiba esse valor atualizado quando as transações MV2PL precisem verificar se certa versão está livre para escrita ou não. Diferente do

MV2PL não é necessário que transações MVOCC verifiquem esse campo, apenas escrevam.

Na operação de confirmação do protocolo MVOCC se torna necessário decrementar o campo *read-cnt* de todas as versões no conjunto de leituras da transação que necessita confirmar, ainda pelo motivo da compatibilidade com o MV2PL. O protocolo MVOCC já mantém esse conjunto de leituras e na operação de confirmação já verifica todas as versões lidas por uma transação para ter a certeza que não seja executada uma leitura suja pelas transações.

A sobrecarga do incremento e do decremento nesse campo no protocolo MVOCC é miníma se comparada ao ganho em ter melhor desempenho unindo os dois protocolos em uma carga de trabalho mista.

Para níveis mais baixos de isolamento é seguido a implementação discutida no trabalho [Larson et al. 2011] que tivemos como base de implementação dos protocolos MVOCC e MV2PL.

5.1. Corretude

Vamos agora provar que o *scheduler* do FLEXMVCC só admite históricos multi-versão 1SR. Usamos como base a abordagem descrita em [Larson et al. 2011] que prova a corretude do MVOCC, bem como a seção 5.5.2 do [Weikum and Vossen 2001] que descreve o MV2PL de maneira mais detalhada e prova que admite apenas históricos multi-versão 1SR. Na nossa prova são usadas as notações e os teoremas da Seção 5.2 de [Bernstein et al. 1987] que são definidos para protocolos multi-versão.

O *scheduler* do FLEXMVCC se comporta como um *scheduler* MVOCC, MV2PL e um período de transição entre os dois protocolos. A corretude do MVOCC e do MV2PL já é abordado no trabalho [Larson et al. 2011].

Em [Larson et al. 2011] é definido que a transação T_x é uma transação confirmada com um timestamp inicial chamado T_{xBegin} e um timestamp final chamado T_{xEnd} e para transações do MVOCC são expostas as seguintes propriedades:

Propriedade 1: Os *timestamps* são designados em ordem crescente e monotonicamente, e cada transação possui um *timestamp* exclusivo de início e fim, de forma que $T_{xBegin} < T_{xEnd}$.

Propriedade 2: Uma determinada versão é válida para o intervalo especificado pelos *timestamps* de início e fim. Existe uma ordem total de versões, denotado por $<<$, para uma determinada tupla, conforme determinado pela ordem dos *timestamps* dos intervalos de validade da versão não sobreposta.

Propriedade 3: A transação T_x lê a última versão confirmada chamada T_{xRead} (onde $T_{xBegin} <= T_{xRead} < T_{xEnd}$) e valida (ou seja, repete) a leitura da última versão confirmada a partir de T_{xEnd}. A transação falhará se as duas leituras retornarem versões diferentes.

Propriedade 4: Primeiro verificar a visibilidade de uma versão V, para atualizar ou apagar V. Verificar a visibilidade de V é equivalente a leitura de V. Portanto, uma gravação é sempre precedida por uma leitura: se a transação T_x gravar V_{new}, a transação T_x terá que ler primeiro o V_{old}, onde V_{old} « V_{new}. Além disso, não existe nenhuma versão

V tal que $V_{old} << V << V_{new}$, caso contrário T_x nunca teria confirmado.

Propriedade 5: A transação T_x grava logicamente no *timestamp* T_{xEnd}, porque a versão é invisível para outras transações até o *timestamp* T_{xEnd}.

Com base em [Weikum and Vossen 2001] expomos as propriedades para transações MV2PL:

Propriedade 6: Dado uma transação T_x e uma T_y, T_x só poderá ler uma versão V, caso a transação T_y que criou a versão V já tenha confirmado.

Propriedade 7: Verificar a visibilidade de uma versão V primeiro, para atualizar ou apagar V. Verificar a visibilidade de V é equivalente a leitura de V. Portanto, uma gravação é sempre precedida por uma leitura: se a transação T_x gravar V_{new}, a transação T_x terá que ler primeiro o V_{old}, onde $V_{old} << V_{new}$. Além disso, não existe nenhuma versão V tal que $V_{old} << V << V_{new}$, caso contrário T_x nunca teria escrito V_{new}.

Propriedade 8: Operações de leitura são sempre direcionadas à versão atual, e bloqueio de confirmação de escritores concorrentes, que são incompatíveis com os bloqueios de leitura, servem para determinar situações na qual uma nova versão confirmada é produzida enquanto um leitor ainda está em andamento. Nesse caso, a confirmação do escritor é postergada até que o leitor tenha terminado.

As propriedades **1**, **2** e **5** também são propriedades das transações MV2PL.

O grafo de serialização multi-versão MVSG $(H, <<)$ é um grafo definido em um histórico multi-versão H e uma ordem total da versão $<<$. O MVSG possui nós para as transações confirmadas em H e, por definição, existe uma aresta $T_i \rightarrow T_j$ no MVSG (onde i, j, k são distintos), se e somente se:

A) T_i **escreve** V_i **e** T_j **lê** V_i

ou

B) T_i **escreve** V_i **e** T_k **lê** V_j**, onde** $V_i << V_j$

ou

C) T_i **lê** V_k **e** T_j **escreve** V_j**, onde** $V_k << V_j$

Vamos provar que todas as arestas do MVSG são ordenadas com relação à ordem do *timestamp* final das transações MVOCC e MV2PL misturadas. Ou seja, provaremos que qualquer aresta direcionada $T_i \rightarrow T_j$ sempre apontará de uma transação T_i para uma transação T_j tal que $T_{iEnd} < T_{jEnd}$.

Seja, T_o uma transação MVOCC que gera versões V_o e T_p uma transação MV2PL que gera versão V_p. Temos que:

A.1) T_p **escreve** V_p **e** T_o **lê** V_p**:**

Para (A.1), deixe T_o ler V_p em $T_{oRead} < T_{oEnd}$. Se $T_{oRead} < T_{pEnd}$, das propriedades 3 e 5, teria sido impossível ler V_p, como não é confirmado. Portanto, $T_{pEnd} < T_{oRead} < T_{oEnd}$, então, a aresta $T_p \rightarrow T_o$ é ordenada com relação ao *timestamp* final, como $T_{pEnd} < T_{oEnd}$.

A.2) T_o **escreve** V_o **e** T_p **lê** V_o

Para (A.2), a mesma discussão em (A.1) é valida, com a troca da propriedade 3 exclusiva do MVOCC pela propriedade 6 exclusiva do MV2PL.

B.1) T_o **escreve** V_o **e** T_k **lê** V_p**, onde** $V_o << V_p$

B.1.1) se T_k **for uma transação MVOCC:**

Para (B.1.1), T_k lê V_p, portanto, a leitura é precedida por T_p escrevendo V_p e confirmando. Além disso, da propriedade 3, $T_{pEnd} < T_{kRead}$ ou V_p não seria visível no T_{kRead}. Da propriedade 7, desde que T_p escreveu V_p, isso significa que T_p leu V_o (ou qualquer versão posterior) em T_{pRead}, onde $T_{pRead} < T_{pEnd}$. Portanto, a partir de (A.2), $T_{oEnd} < T_{pRead} < T_{pEnd}$, a aresta $T_o \rightarrow T_p$ é ordenada com relação ao *timestamp* final, como $T_{oEnd} < T_{pEnd}$.

B.1.2) se T_k **for uma transação MV2PL:**

Para (B.1.2), a mesma discussão em (B.1.1) é valida, com a troca da propriedade 3 exclusiva do MVOCC pela propriedade 6 exclusiva do MV2PL.

B.2) T_p **escreve** V_p **e** T_k **lê** V_o**, onde** $V_p << V_o$

B.2.1) se T_k **for uma transação MVOCC:**

Para (B.2.1), a mesma discussão em (B.1.1) é valida, com a troca da propriedade 7 exclusiva do MV2PL pela propriedade 4 exclusiva do MVOCC.

B.2.2) se T_k **for uma transação MV2PL:**

Para (B.2.2), a mesma discussão em (B.1.2) é valida, com a troca da propriedade 7 exclusiva do MV2PL pela propriedade 4 exclusiva do MVOCC.

C.1) T_o **lê** V_k **e** T_p **escreve** V_p**, onde** $V_k << V_p$

Para (C.1), deixe a T_o ler V_k no T_{oRead}, onde $T_{oRead} < T_{oEnd}$. T_p escreve V_p no T_{pEnd}. Existem três casos:

C.1.1) $T_{pEnd} < T_{oRead} < T_{oEnd}$. Isso viola a propriedade 3, já que V_p era uma versão confirmada no T_{oRead}, portanto, T_o teria lido V_p, não V_k.

C.1.2) $T_{oRead} < T_{pEnd} < T_{oEnd}$. Isso viola a propriedade 3, pois a T_o repetiria a leitura no T_{oEnd} durante a validação e leria V_p. No entanto, T_o leu V_k na T_{oRead}, portanto, T_o falharia na validação e nunca participaria do MVSG.

C.1.3) $T_{oRead} < T_{oEnd} < T_{pEnd}$. T_o validaria a leitura em T_{oEnd}, leria V_k novamente e confirmaria. Portanto, a aresta $T_o \rightarrow T_p$ é ordenada com relação ao *timestamp* final, como $T_{oEnd} < T_{pEnd}$.

C.2) T_p **lê** V_k **e** T_o **escreve** V_o**, onde** $V_k << V_o$

Para (C.2), deixe a T_p ler V_k no T_{pRead}, onde $T_{pRead} < T_{pEnd}$. T_o escreve V_o no T_{oEnd}. Existem três casos:

C.2.1) a mesma discussão em (C.1.1) é valida, com a troca da propriedade 3 exclusiva do MVOCC pela propriedade 6 exclusiva do MV2PL.

C.2.2) $T_{pRead} < T_{oEnd} < T_{pEnd}$. Isso viola a propriedade 6 e 8, pois A T_p leria a versão V_k e pela a propriedade 8 T_o não poderia confirmar suas alterações até que o T_p finalizasse, portanto T_o falharia na confirmação e nunca participaria do MVSG.

C.2.3) a mesma discussão em (C.1.3) é valida aqui.

Portanto, de (A) e (B), todas as arestas no MVSG são ordenadas com relação à ordem do *timestamp* final da transação e, portanto (da Propriedade 1), elas não podem ser envolvidas em um ciclo. Daí se conclui que os históricos de MV que são aceitos pelo nosso *scheduler* de controle de concorrência FLEXMVCC são 1SR.

6. Experimentação

Apresentamos agora nossa análise experimental do protocolo de controle de concorrência discutido neste artigo. O protocolo proposto foi implementado no SGBD Peloton [Pavlo et al. 2017], que armazena tuplas em heaps em memória não ordenados, orientados a linhas. O Peloton, como um SGBD em memória de código aberto, foi escolhido por utilizar controle de concorrência multi-versão. Nele, implementamos o FLEXMVCC, além dos protocolos MVOCC e MV2PL. As transações foram executadas sob o nível de isolamento SERIALIZABLE.

6.1. Configuração

O build do Peloton foi feito em uma máquina Intel Core i7-7800X com 6 núcleos e 12 threads rodando a uma frequência base de 3.5GHz e 128GB de memória SDRAM DDR4, com sistema operacional Ubuntu 16.04.3.

Utilizamos o benchmark YCSB [Cooper et al. 2010] em nossa experimentação por sua flexibilidade ao modelar diferentes configurações de workloads de aplicações OLTP. O framework OLTPBench [Difallah et al. 2013] foi utilizado na experimentação por sua facilidade na configuração e execução do YCSB, além da coleta de informações como latência e vazão por tipo de transação. O banco de dados contém uma única tabela com 50 mil tuplas, cada uma com 11 atributos inteiros de 64 bits.

6.2. Cargas de trabalho

Selecionamos quatro cargas de trabalho para variar o número de transações com operações de read e update: (1) 80% leituras, 20% updates, (2) 60% leituras, 40% updates, (3) 40% leituras, 60% updates e (4) 20% leituras, 80% updates. As cargas de trabalho citadas foram selecionadas devido ao favorecimento do protocolo MVOCC em relação à operações de leitura e ao MV2PL em relação a operações de escrita. Estas cargas foram executadas em sequência, em cada protocolo: MVOCC, MV2PL e FLEXMVCC. O benchmark foi configurado para utilizar 50 threads concorrentes onde cada transação executa 10 operações. Esta modificação foi feita para aumentar o tempo de duração de cada transação, consequentemente, auxiliando a avaliação dos respectivos protocolos de controle de concorrência.

Nos nossos experimentos, a carga (1) apresenta maior vazão ao ser executada pelo MVOCC, a carga (2) pelo MV2PL e assim sucessivamente. O FLEXMVCC, durante essa execução sequencial, foi modificado para trocar o protocolo dependendo da configuração da carga, de forma que as transações executadas utilizariam o protocolo mais otimizado para o tipo de carga no momento.

6.3. Resultados Experimentais

A Figura 5 mostra o resultado da execução do experimento, contendo as médias de vazão das quatro cargas de trabalho em cada protocolo. Observamos que o FLEXMVCC

obtém maior vazão do que os outros protocolos, devido à sua flexibilidade em alternar os protocolos conforme a mudança no perfil da carga de trabalho.

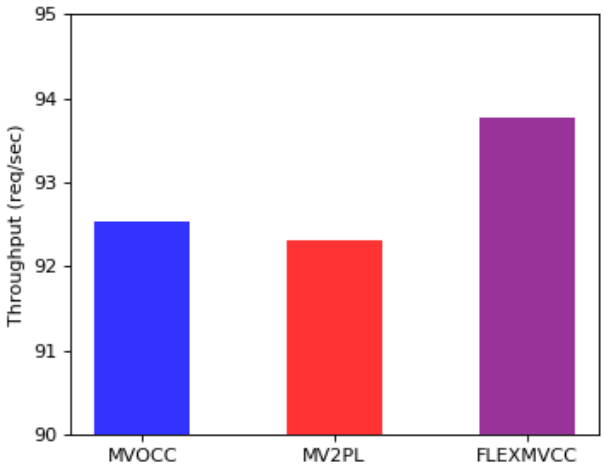


Figura 5. Média da vazão dos protocolos executando as cargas (1), (2), (3) e (4) sequencialmente

7. Conclusão

Este trabalho apresenta o FLEXMVCC, um protocolo de controle de concorrência que une dois protocolos de controle de concorrência multi-versão compatíveis entre si, o MVOCC e o MV2PL. Alternando entre os dois, o FLEXMVCC consegue se adaptar às mudanças de perfil na carga de trabalho, conferindo um aumento no desempenho do banco de dados, evidenciado pelo acréscimo da vazão observado por meio de benchmarks. Também provamos a corretude do FLEXMVCC, principalmente no período de transição entre o MVOCC e o MV2PL, característica essencial quando é proposto um novo mecanismo de controle de concorrência.

Agradecimentos

Esta pesquisa foi apoiada pelo Laboratório de Sistemas e Banco de Dados e FUNCAP (Bolsa BMD-0008-01237.01.09/17).

Referências

Bernstein, P. A., Hadzilacos, V., and Goodman, N. (1987). Concurrency control and recovery in database systems.

Cooper, B. F., Silberstein, A., Tam, E., Ramakrishnan, R., and Sears, R. (2010). Benchmarking cloud serving systems with ycsb. In *Proceedings of the 1st ACM Symposium on Cloud Computing*, SoCC '10, pages 143–154, New York, NY, USA. ACM.

Diaconu, C., Freedman, C., Ismert, E., Larson, P.-A., Mittal, P., Stonecipher, R., Verma, N., and Zwilling, M. (2013). Hekaton: Sql server's memory-optimized oltp engine. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, pages 1243–1254. ACM.

Difallah, D. E., Pavlo, A., Curino, C., and Cudre-Mauroux, P. (2013). Oltp-bench: An extensible testbed for benchmarking relational databases. *Proc. VLDB Endow.*, 7(4):277–288.

Harizopoulos, S., Abadi, D. J., Madden, S., and Stonebraker, M. (2008). Oltp through the looking glass, and what we found there. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 981–992. ACM.

Kung, H.-T. and Robinson, J. T. (1981). On optimistic methods for concurrency control. *ACM Transactions on Database Systems (TODS)*, 6(2):213–226.

Larson, P.-Å., Blanas, S., Diaconu, C., Freedman, C., Patel, J. M., and Zwilling, M. (2011). High-performance concurrency control mechanisms for main-memory databases. *Proceedings of the VLDB Endowment*, 5(4):298–309.

Narula, N., Cutler, C., Kohler, E., and Morris, R. (2014). Phase reconciliation for contended in-memory transactions. In *OSDI*, volume 14, pages 511–524.

Pavlo, A., Angulo, G., Arulraj, J., Lin, H., Lin, J., Ma, L., Menon, P., Mowry, T. C., Perron, M., Quah, I., Santurkar, S., Tomasic, A., Toor, S., Aken, D. V., Wang, Z., Wu, Y., Xian, R., and Zhang, T. (2017). Self-driving database management systems. In *CIDR*. www.cidrdb.org.

Shang, Z., Li, F., Yu, J. X., Zhang, Z., and Cheng, H. (2016). Graph analytics through fine-grained parallelism. In *Proceedings of the 2016 International Conference on Management of Data*, pages 463–478. ACM.

Su, C., Crooks, N., Ding, C., Alvisi, L., and Xie, C. (2017). Bringing modular concurrency control to the next level. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 283–297. ACM.

Tanger, D. (2017). Toward coordination-free and reconfigurable mixed concorrency control. Master's thesis, University of Chicago.

Tu, S., Zheng, W., Kohler, E., Liskov, B., and Madden, S. (2013). Speedy transactions in multicore in-memory databases. In *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles*, pages 18–32. ACM.

Wang, T. and Kimura, H. (2016). Mostly-optimistic concurrency control for highly contended dynamic workloads on a thousand cores. *Proceedings of the VLDB Endowment*, 10(2):49–60.

Weikum, G. and Vossen, G. (2001). *Transactional information systems: theory, algorithms, and the practice of concurrency control and recovery*. Elsevier.

Xie, C., Su, C., Littley, C., Alvisi, L., Kapritsos, M., and Wang, Y. (2015). High-performance acid via modular concurrency control. In *Proceedings of the 25th Symposium on Operating Systems Principles*, pages 279–294. ACM.

Yu, X. (2015). *An evaluation of concurrency control with one thousand cores*. PhD thesis, Massachusetts Institute of Technology.

Yu, X., Pavlo, A., Sanchez, D., and Devadas, S. (2016). Tictoc: Time traveling optimistic concurrency control. In *Proceedings of the 2016 International Conference on Management of Data*, pages 1629–1642. ACM.

SBBD 2018

Short, Vision and Industrial Papers

Rumo à Integração da Álgebra de Workflows com o Processamento de Consulta Relacional*

João Antonio Ferreira[1], Jorge Soares[1], Fabio Porto[2], Esther Pacitti[3], Rafaelli Coutinho[1], Eduardo Ogasawara[1]

[1]CEFET/RJ - Centro Federal de Educação Tecnológica Celso Suckow da Fonseca

[2]LNCC - Laboratório Nacional de Computação Científica

[3]Inria & University of Montpellier

joao.parana@acm.org, jorge@eic.cefet-rj.br,fporto@lncc.br,

Esther.Pacitti@inria.fr,rafaelli.coutinho@cefet-rj.br, eogasawara@ieee.org

Abstract. *Workflows emerged as a basic abstraction for structuring data analysis experiments in the current Data Intensive Scalable Computing (DISC) scenario. In many situations, these workflows are intensive, either computationally or in relation to data management, requiring execution in high-performance processing environments. However, parallelizing the execution of workflows commonly requires laborious programming, in an ad hoc manner and in a low level of abstraction, which makes it difficult to explore optimization opportunities. Some algebraic approaches have been developed to mitigate such limitation. This work moves in the direction converging the workflow algebra with relational query processing.*

Resumo. *Os workflows emergiram como uma abstração básica para estruturar experimentos de análise de dados no atual cenário de DISC (*Data Intensive Scalable Computing*). Em muitas situações, estes workflows são intensivos, seja computacionalmente ou em relação à manipulação de dados, exigindo a execução em ambientes de processamento de alto desempenho. Entretanto, paralelizar a execução de workflows comumente requer programação trabalhosa, de modo ad hoc e em baixo nível de abstração, o que torna difícil a exploração das oportunidades de otimização. Algumas abordagens algébricas foram desenvolvidas visando mitigar tal limitação. Este trabalho caminha na direção de convergir a álgebra de workflows com o processamento de consultas relacionais.*

1. Contexto

Apesar de alguns sistemas de workflows possuírem recursos para execução paralela, paralelizar um workflow de larga escala é uma tarefa difícil, *ad hoc* e trabalhosa. Na maioria das soluções existentes, cabe aos usuários dos sistemas decidirem a ordem e as dependências entre as atividades além das estratégias de paralelização. Estas decisões, em muitos casos, restringem as oportunidades de otimização da execução do workflow que poderiam levar a melhorias significativas de desempenho [Ogasawara et al., 2011], principalmente

*Os autores agradecem à FAPERJ, à CAPES e ao CNPq pelo financiamento parcial do projeto.

quando tais workflows são expressos por funções definidas por usuário (UDF) [Rheinländer et al., 2017].

A adoção de uma abordagem algébrica para especificar workflows vem sendo amplamente estudada e permite realizar a otimização da execução paralela de workflows de modo sistemático [Ogasawara et al., 2011; Fegaras, 2017], *i.e.*, de modo que o desenvolvedor não tenha que se preocupar com codificações paralelas, mas tão somente com a especificação do workflow considerando informação do domínio [Liu et al., 2015; Rheinlander et al., 2015].

Na álgebra de workflows, os dados são uniformemente representados por meio de relações e as atividades são regidas por operações algébricas que possuem semântica sobre a produção e o consumo dos dados. Considera-se, nesta álgebra, que atividades consomem e produzem relações. Isso traz uma uniformidade no modelo de dados e possibilita a geração de workflows prontos para execução em paralelo. As relações são definidas como um conjunto de tuplas de dados primitivos (*i.e.*, inteiro, real, alfanumérico, data, etc) ou referência a arquivo (via URI: *Uniform Resource Identifier*). No que tange ao tratamento de arquivos, o seu formato é considerado uma caixa preta. Desta forma, tem-se uma uniformização do tratamento dos arquivos, sejam eles textuais, semiestruturados ou binários. Assim, é de responsabilidade das atividades no workflow saber consumir ou produzir estes dados [Ogasawara et al., 2011].

Cada relação R possui um esquema $\mathcal{R}$ e pode ser especificada como $R(\mathcal{R})$. Dada uma relação $R(\mathcal{R})$, representa-se $atr(R)$ como um conjunto de atributos de R e $key(R)$ como o conjunto contendo os atributos chave de R. De modo análogo à álgebra relacional, relações podem ser manipuladas por operações de conjunto: união ($\cup$), interseção ($\cap$) e diferença ($-$), desde que os seus esquemas sejam compatíveis (aridade da relação e domínio de cada atributo). Pode-se atribuir uma relação a variáveis de relação para posterior reuso usando a atribuição $\leftarrow$ (ex.: $T \leftarrow R_1 \cup R_2$) [Elmasri and Navathe, 2015].

No escopo deste trabalho, uma atividade compreende uma UDF (encapsulando a invocação de um programa ou a execução de uma expressão da álgebra relacional) e esquemas de relações, tanto de entrada quanto de saída. As atividades do workflow são regidas por operações algébricas que especificam a razão de consumo e produção entre as tuplas. Esta característica possibilita um tratamento uniforme para as atividades, viabilizando a realização de transformações algébricas. A álgebra inclui seis operações (resumidas na Tabela 1): Map, $SplitMap$, $Reduce$, $Filter$, $SRQuery$ e $MRQuery$. A maioria das operações algébricas consome uma única relação, com exceção da operação $MRQuery$ que consome uma sequência de relações. As primeiras quatro operações são usadas para apoiar atividades que executam programas encapsulados por meio de UDF. As duas últimas operações são usadas para executar atividades que processam expressões de álgebra relacional. Isso significa que a UDF deve ser compatível com a operação da atividade em termos de interface para consumo e produção de dados [Hsu et al., 2010].

Neste contexto, os workflows podem ser otimizados para execução paralela por meio de transformações algébricas. Cada transformação aplicada, apesar de garantir que o workflow produza o mesmo resultado, traz uma diferença em termos de custo computacional. Em outras palavras, expressões algébricas equivalentes produzem diferentes planos de execução do workflow. Pode-se avaliar o custo destes planos por meio de uma

Tabela 1. Resumo das operações algébricas

Operação	UDF	Operandos adicionais	Resultado	Cons. × Prod. tuplas
Map	programa	Relação R	Relação S	$1:1\ por\ \|R\|$
SplitMap	programa	Relação R	Relação S	$1:m\ por\ \|R\|$
Reduce	programa	Relação R e atrs	Relação S	$n:1\ por\ \|\pi_{atrs}\|$
Filter	programa	Relação R	Relação S	$1:(0-1)\ por\ \|R\|$
SRQuery	exp. relacional	Relação R	Relação S	$n:m$
MRQuery	exp. relacional	Relações $(R_1 \cdots R_i)$	Relação S	$(n_1 \cdots n_i):m$

função de custo. Desta forma, a abordagem algébrica possibilita a visualização do problema de execução paralela de workflow de modo análogo à otimização de consultas em bancos de dados relacionais.

2. Otimização de Workflows

Considere um workflow representado em uma extensão de *XML Process Definition Language* (XPDL)[1] e visualmente apresentado pelo grafo da Figura 1.a. Tal representação tem uma correspondência direta com as expressões da álgebra de workflows descritas na Figura 1.b. Cada relação intermediária tem um esquema específico (no exemplo foram deixados parecidos apenas para facilitar o entendimento).

A semântica presente nas operações algébricas possibilita a reescrita de workflows, gerando expressões equivalentes, porém, com desempenho de paralelismo distinto por meio de um processo de otimização de workflow (Figura 1.c). A navegação por esse espaço de solução, formado por expressões equivalentes de workflows, permite a busca por expressões eficientes. Essa busca, orientada por um modelo de custos de execução paralela, viabiliza, assim, a otimização da execução do workflow.

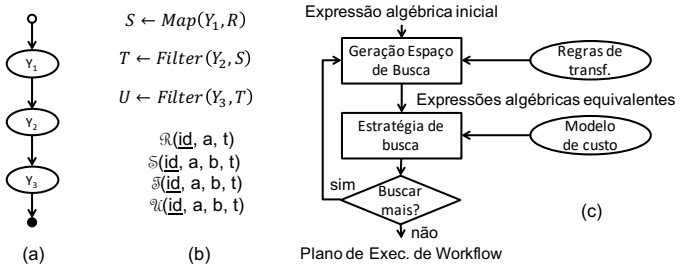


Figura 1. Workflow representado em grafo (a), Workflow representado em álgebra de workflows (b), Processo de otimização de execução de workflows (c)

No escopo deste trabalho considera-se o modelo de execução constituído por uma ativação de atividades e por uma forma simplificada de execução destas ativações, baseadas em distribuição estática de pipelines [Ogasawara et al., 2011], normalmente presentes em sistemas *Data Intensive Scalable Computing* (DISC) [Bryant, 2011], como no caso do Spark [Zaharia et al., 2016]. Uma ativação é um objeto autocontido que possui as informações necessárias para a execução de uma instância da atividade em qualquer um dos núcleos disponíveis [Ogasawara et al., 2011]. Desta forma, no tocante à execução, a

[1] A forma de representar workflows em XPDL pode ser vista em Ogasawara et al. [2013]

ativação contém as informações relacionadas à UDF, os dados (tuplas a serem consumidas e, após a execução da UDF, as tuplas produzidas) e o status de execução. A ativação é inspirada no conceito de ativação em banco de dados [Bouganim et al., 1996]. As ativações podem consumir e produzir conjunto de tuplas; entretanto, uma ativação contém a unidade mínima de dados necessários para que, ainda assim, a execução da instância de uma atividade seja realizada. A razão entre o consumo e a produção de tuplas em uma ativação varia de acordo com a operação algébrica que rege aquela atividade (Tabela 1). A saída de uma atividade é a união de todas as tuplas produzidas por suas ativações.

As execuções das ativações podem ser divididas em três etapas: instrumentação da entrada, invocação de programa e extração de saída. A instrumentação da entrada extrai os valores das tuplas de entrada e prepara para a execução do programa, configurando os valores das entradas dos parâmetros de acordo com os tipos de dados esperados. A invocação de programa despacha e monitora o programa associado à atividade. A extração de saída coleta os dados da saída do programa executado, transformando-os nas tuplas de saída.

Ignorando-se neste momento o modelo de execução (materialização *versus* pipeline) [Elmasri and Navathe, 2015], define-se como otimização de workflow, o processo a partir do qual se exploram as possíveis configurações para se obter uma que minimize o custo computacional. A Figura 1.c descreve o processo de otimização de workflow, que remonta ao processo tradicional de otimização de consultas em banco de dados, no qual foca-se na identificação de um plano de execução de consulta que minimize uma função de custo. A otimização de consultas tipicamente restringe o tamanho do espaço de busca que se pode considerar, por meio de heurísticas que são comumente aplicadas, como as que realizam as operações de projeção e seleção nas relações base de modo a minimizar o número de tuplas intermediárias antes das operações de junções. No modelo relacional, as permutações de uma árvore de junção são comumente as mais importantes para influenciar as consultas relacionais [Tamer Ozsu and Valduriez, 2011], particularmente focando na redução de tuplas das relações intermediárias. Por conta disto, assume-se a aplicação de heurísticas que se concentram na otimização das árvores de junções. Nesta perspectiva, a otimização de workflows proposta neste trabalho possui várias semelhanças em relação às tradicionais otimizações de consultas em banco de dados. O maior objetivo da otimização de workflows é reduzir o tamanho das relações intermediárias que servem como entrada para atividades que invocam UDF com custo computacional elevado[2]. Assim, de modo análogo ao processamento de consultas, no qual a ordem das junções direciona a otimização, na abordagem algébrica para workflows a ordem de execução das UDF direciona a otimização de workflows.

Cabe salientar que as dependências em workflows impõem restrições às transformações algébricas que podem ser aplicadas durante o processo de otimização. Por exemplo, se uma atividade Y_i produz dados que são consumidos por atividades Y_j, a atividade Y_j não pode ser posicionada em uma ordem que anteceda a atividade Y_i. Sempre que possível, antecipa-se a execução de atividades que reduzam a quantidade de tuplas nas relações intermediárias e, correspondentemente, adiam-se as atividades que produzem mais dados.

[2]O custo computacional da execução de uma UDF pode ser conhecido por meio de proveniência

Este cenário de analogia torna-se mais interessante quando se consegue uniformizar o modelo relacional com o de workflows por meio de transformações algébricas. Tais ações possibilitam usufruir do conhecimento de otimização de consultas no contexto de workflows. Desta forma, este trabalho aponta potenciais benefícios ao se formalizar o mapeamento da álgebra de workflows na álgebra relacional. Tal abordagem possibilita elaborar um processador de workflows que tenha o modelo de otimização de consultas como um dos seus alicerces. A Tabela 2 apresenta a relação entre álgebra de workflows e as UDF invocadas nos workflows, e a sua tradução para expressões da álgebra relacional. No caso do Map há uma tradução para a invocação da projeção da álgebra relacional. Neste caso, tem-se o mapeamento dos atributos invocados, que por simplicidade de notação está marcado como asterisco na base de tupla a tupla. De modo análogo, o $SplitMap$ foi mapeado para uma projeção especializada ($\overline{\pi}$) apenas para deixar claro que a UDF realiza consumo tupla a tupla, produz um conjunto de tuplas na saída e que este operador estendido deve ser capaz de considerar esta diferença. O operador $Reduce$ é semelhante às UDF de agrupamento de dados. Finalmente, o operador $Filter$, apesar de operar na seleção, também é um operador tupla a tupla, por ser invocado no predicado da seleção.

Tabela 2. Mapeamento da álgebra de workflows para álgebra relacional

Operação	Álgebra de Workflow	Álgebra Relacional
Map	$S \leftarrow Map(Y, R)$	$S \leftarrow \pi_{u_Y(*)}R$
SplitMap	$S \leftarrow SplitMap(Y, R)$	$S \leftarrow \overline{\pi}_{u_Y(*)}R$
Reduce	$S \leftarrow Reduce(Y, atrs, R)$	$S \leftarrow {}_{atrs}\Gamma_{u_Y(*)}R$
Filter	$S \leftarrow Filter(Y, R)$	$S \leftarrow \sigma_{u_Y(*)}R$

3. Estudo de caso

Considere a execução do workflow da Figura 1, na sua forma original mostrada em (b): (i) $S \leftarrow Map(Y_1, R)$, (ii) $T \leftarrow Filter(Y_2, S)$ e (iii) $U \leftarrow Filter(Y_3, T)$. Levando em conta o esquema das relações do workflow e traduzindo-se tais expressões para álgebra relacional, tem-se: (i) $S \leftarrow \pi_{id,a,b,t}(\pi_{U_{Y_1}(id,a,t)}R)$, (ii) $T \leftarrow \pi_{id,a,b,t}(\sigma_{U_{Y_2}(id,a,b,t)}S)$ e (iii) $U \leftarrow \pi_{id,a,b,t}(\sigma_{U_{Y_3}(id,a,b,t)}T)$.

Com a introdução da semântica dos atributos consumidos e produzidos pelas UDF e a relação de tuplas produzidas nestes contextos, pode-se aplicar as transformações algébricas conhecidas e escrever: $U \leftarrow \pi_{id,a,b,t}(\sigma_{U_{Y_2}(id,a,b,t)}(\sigma_{U_{Y_3}(id,a,b,t)}S))$, retirando-se a relação intermediária T (formando-se um pipeline). Pode-se, até mesmo, inverter a ordem das operações $W \leftarrow \pi_{id,a,b,t}(\sigma_{U_{Y_3}(id,a,b,t)}S)$ e $U \leftarrow \pi_{id,a,b,t}(\sigma_{U_{Y_2}(id,a,b,t)}W)$, materializando-se intermediariamente a relação W antes de produzir U. Note que estas escolhas passam a ser viáveis de serem feitas devido a uniformização da álgebra de workflows com a álgebra relacional e a computação das complexidades por meio de funções de custo. Além disto, esta representação, de modo análogo ao relacional, viabiliza as decisões de modelo de execução (pipeline *versus* materialização) independentemente da especificação do workflow pelos especialistas do domínio.

Foi feita uma avaliação experimental preliminar por meio de dados sintéticos a partir do workflow da Figura 1.b usando o Chiron [Ogasawara et al., 2013]. Considerou-se que a atividade Y_2 apresentava seletividade de 100% (todas as tuplas são aceitas), que

ela é computacionalmente intensiva e que ao mesmo tempo, a atividade Y_3 variava o seu fator de seletividade de 20% a 100% nas diferentes execuções do workflow. A alteração da ordem das atividades Y_2 para Y_3 se mostrou benéfica. Observe que a razão da seletividade variar vem do fato das múltiplas execuções do workflow (sensibilizado pelo atributo b calculado na atividade Y_1). A versão otimizada traz vantagens quando o fator de seletividade de Y_3 é baixo. Este experimento foi executado em um *cluster* com 64 núcleos, com número de tuplas inicial igual a 16.384 e custo médio de execução da atividade Y_1, Y_2 e Y_3, respectivamente, iguais a 1.0, 4.0 e 0.25 segundos. Desta forma, houve uma melhoria no tempo de execução que passou de 22.4 para 8.7 minutos, trazendo uma redução de mais de 60%. Em cenário de larga escala, pode-se considerar que tal abordagem traz possibilidades de avanços na execução dos workflows.

Como trabalhos futuros, pretende-se formalizar de modo mais rigoroso o comportamento das expressões algébricas e as diferentes UDF provinda das diferentes operações da álgebra de workflows. Pretende-se conduzir testes de desempenho de escalabilidade, mostrando tanto volumes de dados em baixa escala quanto em larga escala e discutir o custo computacional envolvido.

Referências

Bouganim, L., Florescu, D., and Valduriez, P. (1996). Dynamic Load Balancing in Hierarchical Parallel Database Systems. In *Proceedings of the 22th International Conference on Very Large Data Bases*, VLDB '96, pages 436–447, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Bryant, R. (2011). Data-intensive scalable computing for scientific applications. *Computing in Science and Engineering*, 13(6):25–33.

Elmasri, R. and Navathe, S. B. (2015). *Fundamentals of Database Systems*. Pearson, Boston und 24 andere, 7 edition.

Fegaras, L. (2017). An algebra for distributed Big Data analytics. *Journal of Functional Programming*.

Hsu, M., Chen, Q., Wu, R., Zhang, B., and Zeller, H. (2010). Generalized UDF for analytics inside database engine. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6184 LNCS:742–754.

Liu, J., Pacitti, E., Valduriez, P., and Mattoso, M. (2015). A Survey of Data-Intensive Scientific Workflow Management. *Journal of Grid Computing*, pages 1–37.

Ogasawara, E., de Oliveira, D., Valduriez, P., Dias, J., Porto, F., and Mattoso, M. (2011). An algebraic approach for data-centric scientific workflows. In *Proceedings of the VLDB Endowment*, volume 4, pages 1328–1339.

Ogasawara, E., Dias, J., Silva, V., Chirigati, F., Oliveira, D. d., Porto, F., Valduriez, P., and Mattoso, M. (2013). Chiron: a parallel engine for algebraic scientific workflows. *Concurrency and Computation: Practice and Experience*, 25(16):2327–2341.

Rheinlander, A., Heise, A., Hueske, F., Leser, U., and Naumann, F. (2015). SOFA: An extensible logical optimizer for UDF-heavy data flows. *Information Systems*, 52:96–125.

Rheinländer, A., Leser, U., and Graefe, G. (2017). Optimization of complex dataflows with user-defined functions. *ACM Computing Surveys*, 50(3).

Tamer Ozsu, M. and Valduriez, P. (2011). *Principles of Distributed Database Systems*. Springer, New York, 3 edition.

Zaharia, M., Franklin, M. J., Ghodsi, A., Gonzalez, J., Shenker, S., Stoica, I., Xin, R. S., Wendell, P., Das, T., Armbrust, M., Dave, A., Meng, X., Rosen, J., and Venkataraman, S. (2016). Apache spark: A unified engine for big data processing. *Communications of the ACM*, 59(11):56–65.

$FReeP$: towards parameter recommendation in scientific workflows using preference learning*

Daniel Silva Jr.[1], **Aline Paes**[1], **Esther Pacitti**[2], **Daniel de Oliveira**[1]

[1]Department of Computer Science, Universidade Federal Fluminense - Brazil

danieljunior@id.uff.br, {alinepaes,danielcmo}@ic.uff.br

[2]Inria, LIRMM and University of Montpellier - France

Esther.Pacitti@lirmm.fr

Abstract. *Scientific workflows are a de facto standard for modeling scientific experiments. However, several workflows have too many parameters to be manually configured. Poor choices of parameter values may lead to unsuccessful executions of the workflow. In this paper, we present $FReeP$, a parameter recommendation algorithm that suggests a value to a parameter that agrees with the user preferences. $FReeP$ is based on the Preference Learning technique. A preliminary experimental evaluation performed over the SciPhy workflow showed the feasibility of $FReeP$ to recommend parameter values for scientific workflows.*

1. Introduction

Scientific workflows are considered a *de facto* standard for modeling scientific experiments that are compute- and data-intensive [Zhao et al. 2008]. They are abstractions that represent the flow of data among activities (*i.e.,* program invocations). The Scientific Workflow Management Systems (SWfMS) are responsible for managing the execution of workflows and collecting provenance data [Freire et al. 2008], which represent the execution history of the workflow. A workflow can be formally defined as a directed acyclic graph $W(A, Dep)$. The nodes $A = \{a_1, a_2, ..., a_n\}$ are the activities and the edges Dep represent the data dependencies among activities in A. Thus, given $a_i \mid (1 \leq i \leq n)$, the set $P = \{p_1, p_2, .., p_m\}$ represents the possible input parameters for activity a_i that define the behavior of a_i.

Scientific Workflows are applied in many fields, such as biology and astronomy. Given the increasingly complexity of experiments in these domains [Zhao et al. 2008], many workflows have many parameters to be configured by users (*e.g.,* > 40). The configuration of such parameters is a sensitive point because it can impact the execution time of the workflow and the usefulness of the produced results. This way, it is required that the user is able to configure the workflow (*e.g.,* setting values for all the m parameters of an activity a_i) as best as possible. Let us take as an example the workflow SciPhy [Ocaña et al. 2011]. SciPhy aims at generating phylogenetic trees (*i.e.,* trees that represent the evolutionary history of an organism). It is composed of four activities: (i) sequence alignment; (ii) conversion of alignment format; (iii) search for the best evolutionary model; and (iv) construction of the phylogenetic tree. Although conceptually simple (*i.e.,* it has only four activities), SciPhy can be complex to be configured because of the number of parameters that should be explored. In addition, the user does not necessarily know *a priori* which configuration generates the best-quality phylogenetic tree. For example, each input file that contains DNA or RNA sequences has a number

*Authors would like to thank FAPERJ, CAPES and CNPq for partially sponsoring this research

of sequences num_seq and a maximum length of sequence $length\_seq$, which are input parameters of activity (i). However, to activity (iv) the input parameter $bootstrap\_replicate$ has its utility limited depending on the choice of values for $length\_seq$. A poor choice of such values can generate a worthless result (in addition to the loss of time and resources).

Thus, it is interesting that the parameter values can be *automatically* recommended to users by a Recommendation System (RS) to increase the chances of producing good results. For example, if in a given SciPhy execution the user sets the value of $num\_seq = 100$, it would be better that the value of $bootstrap\_replicate$ to be recommended by an RS followed the value of num_seq in order to avoid incompatibility of the parameter values. This type of recommendation is feasible, since the SWfMSs collect provenance data, but it is not simple to be performed. Provenance can be used for future recommendations as we know which executions produced successful results and which parameter values were used. The motivation of this paper is, therefore, to recommend parameter values of workflow activities using provenance data collected in previous workflow executions. Thus, we propose a parameter recommendation algorithm named $FReeP$ for scientific workflows that benefits from the other chosen parameters. This way, $FReeP$ is able to suggest a value to a set of parameters that agrees with the user preferences. To accommodate such preferences together with the more appropriate recommendation, we follow a Preference Learning [Fürnkranz and Hüllermeier 2011] technique. The experimental evaluation performed with the SciPhy workflow showed the feasibility of $FReeP$ to recommend parameter values.

2. A Brief on Preference Learning

Recommendation algorithms aim at suggesting the most relevant items to solve a task that requires a choice. In a *personalized* recommendation, each user receives his/her own list of items, based on his/her *preferences*. From the Artificial Intelligence point of view, a preference is an expression of the problem's constraints, but allowing some sort of relaxation [Fürnkranz and Hüllermeier 2011]. In general, Preference Learning consists of inducing a predictive function that, given a set of already established-as-preferred items, it predicts the preferences for a new set of items. The most likely research task in this area is "Learning how to rank", rising from the need of obtaining an ordering relation among the preferences. The ordering task may focus on the class label, directly on the instances, or on a subsect of the objects. Particularly, one of the most used technique to learn preferences is *Pairwise Label Ranking* [Hüllermeier et al. 2008]. It consists of learning the preferences by decomposing the problem in smaller binary preferences problems, *i.e.*, preferences between pairs of classes. Next, an ordering is induced relying on methods that minimize the loss function. This function, in turn, is computed according to the preferences that are violated, considering each different combination of classes.

3. FReeP: Feature Recommender from Preferences

In this paper, we propose a recommendation algorithm named $FReeP$ that relies on provenance data, preference learning, and voting systems to recommend a value for a specified parameter. The recommendation provided by $FReeP$ is the combination of a set of selected recommendations. These recommendations are created according to other specified parameter values. $FReeP$ belongs to the category of Collaborative Filtering [Herlocker et al. 2004] methods and is presented in Algorithm 1.

Algorithm 1 FReeP

Require:

S: { $(param_1^1, val_1^1)$,...$(param_l^m, val_l^m)$ } ▷ l is the number of workflow parameters and m is the number of tuples in the provenance database
F : {attr | attr is a workflow parameter }
C : {attr_preference | attr_preference ∈ F} ▷ *attr_preference* is a workflow parameter where the user defined a value
f(attr) : {preference_value | attr ∈ C} ▷ *preference_value* is the value defined by user for parameter *attr*
P : {(attr,attr_preference) | attr ∈ C ∧ attr_preference ∈ f(attr)} y | y ∈ (F - C)
procedure FREEP(type) ▷ Type is used to select between pure KNN or Label Rank
    $votes \leftarrow \emptyset$; $FS \leftarrow$ POWER_SET$(C) \setminus \emptyset$ ▷ POWERSET it the math operation that returns the set of all subsets
    for each $set \in FS$ **do**
        $horizontal\_partition \leftarrow \emptyset$ ▷ Horizontal partition holds only the instances matching the user preferences
        for each $tuple \in S$ **do** ▷ $tuple$ refers to each tuple in the provenance database
            $flag \leftarrow true$
            for each $(attr, value) \in tuple$ **do**
                if $attr \in C \ \& \ (attr, value) \notin P$ **then**
                    $flag \leftarrow false$
            if $flag$ **then**
                $horizontal\_partition \leftarrow horizontal\_partition \cup \{tuple\}$
        $set \leftarrow set \cup \{y\}$; $vertical\_partition \leftarrow \emptyset$
        for each $tuple \in horizontal\_partition$ **do**
            $filtered\_record\_columns \leftarrow \emptyset$
            for each $(attr, value) \in tuple$ **do**
                if $attr \in set$ **then**
                    $filtered\_record\_columns \leftarrow filtered\_record\_columns \cup (attr, value)$
            $vertical\_partition \leftarrow vertical\_partition \cup filtered\_record\_columns$
        $recommender \leftarrow$ SELECT_RECOMM$(type, vertical_partition)$ ▷ Builds the pure KNN or the KNN + LabelRank
        $to\_recommend \leftarrow \{attr\_preference \mid (attr, attr\_preference) \in P, attr \in set\}$
        $vote \leftarrow$ RECOMMEND$(to_recommend, recommender, type)$ ▷ KNN returns the predicted value, while LabelRank returns the results of the ranking process
        $votes \leftarrow votes \cup \{vote\}$
    $recomendation \leftarrow$ GET_RECOMENDATION$(type, votes)$
    return $recomendation$
function GET_RECOMMENDATION($type$, $votes$)
    if $type =' KNN'$ **then**
        return ARGMAXCOUNT$(votes)$
    else
        return BORDACOUNT$(votes)$
function BORDACOUNT$(rankings)$
    $labels \leftarrow \emptyset$; $labels\_votes \leftarrow \emptyset$
    for each $rank \in rankings$ **do**
        for each $label \in rank$ **do**
            if $label \notin labels$ **then**
                $labels \leftarrow labels \cup label$
                $labels\_votes \leftarrow labels\_votes \cup (label, 0)$
    for each $rank \in rankings$ **do**
        $weight \leftarrow$ LENGTH$(rank) - 1$
        for each $label \in rank$ **do**
            $total\_votes \leftarrow total\_votes \mid (vote, total\_votes) \in labels\_votes, vote == label$
            $new\_total\_votes \leftarrow total\_votes + 2^{weight}$
            $label\_vote \leftarrow (label, new\_total\_votes)$
            $weight \leftarrow weight - 1$
    $recomendation \leftarrow$ ARGMAXLABEL$(labels_votes)$
    return $recomendation$

To perform the parameter recommendation we rely on the tuples extracted from the Provenance database S. These tuples represent successful executions of a given workflow. The set of preferences P, *i.e.,* the parameters for which the scientist already has a set of values, and the parameter to be recommended y are also provided as input to $FReeP$. The first step is to create the set of all subsets of the instances (the power set)[1], grounded by the preferred parameters, called FS. For each subset of FS a horizontal partition is performed over the provenance data, by selecting only the tuples where each parameter (*i.e.,* attribute) has the same value as specified in the user preferences. Next, only the attributes of the tuples that represent the parameters present in the user preferences are gathered (*i.e.,* vertical partition). After that, the algorithm may follow two paths: either it uses a *KNN* classifier [Cover and Hart 1967], or a *LabelRank* [Hüllermeier et al. 2008] classifier. Both of them use the aforementioned partitions to make the recommendation for the parameter y. If the choice is *KNN*, the prediction model is trained with the data in the partition and their respective values for the y parameter, yielding the prediction of values for the other parameter in the partition and stated in the user preferences. On the other hand, if the choice is *LabelRank*, a ranking function is trained in the same way as the *KNN*, however, it returns a sorted list, from the most appropriate value for y to the least one. In this paper, we choose to use the *Pairwise Label Ranking* method to approximate the ranking function.

Each partition generates a possibly different recommendation to the y parameter. This implies in combining the different recommendations in order to arrive at a consensus of what it is the best one. When the chosen method is *KNN*, the recommendation for each partition is a single value, resulting in a list of recommended values. In this case, we use the Simple Voting System [Fishburn 1974] where the value that is most cited among the recommendations is the selected one. On the other hand, the *LabelRank* method returns the recommendations as a *ranking*. The result after the interaction at each partition is a list of *rankings*. Here, we could also employ a simple voting system based on the value ranked as the first one. However, this would neglect other well-ranked values among the different partitions. Thus, we use the Border Count method [Black 1976] that combines all rankings into one, after giving scores for each parameter value according to its position in each ranking. After that, we use the highest ranking value as the final recommendation.

4. Experimental Results

To evaluate $FReeP$, we used the provenance data collected by SciCumulus SWfMS when executing SciPhy workflow [Ocaña et al. 2011]. SciPhy was developed to build phylogenetic trees from DNA, RNA and amino acid sequences. The dataset used to train de Machine Learning models is composed of 376 examples of executions that have not ended at a failure. We follow a 5-Fold Cross Validation procedure [Refaeilzadeh et al. 2016], dividing the examples into 5 disjoint sets, and, at each iteration, 4 sets were used to train the models, while the remaining set was used as test. We considered $K \in [3, 5, 7]$ in both pure KNN and Label Rank methods. We experiment with the recommendation of each parameter of the workflow, separately. We compute the accuracy of both models to evaluate the capability of both approaches in suggesting the right value parameter. Figure 4 shows the experimental results. We can see a clear difference in the accuracies between the recommendation for the parameter num_aligns and all the others. While for the first the recommendation reached values greater than 90%, the others reached accuracies varying from 40% to 60%. This can be explained

[1]In this initial version, we follow the most basic way of choosing the subsets by using the powerset of the original data. This may lead to an exponential number of partitions.

by the smaller variation of values for the parameter num_aligns in the input dataset. Different values of the K parameter did not lead to significant changes in the recommendations, with low standard deviation among the folds.

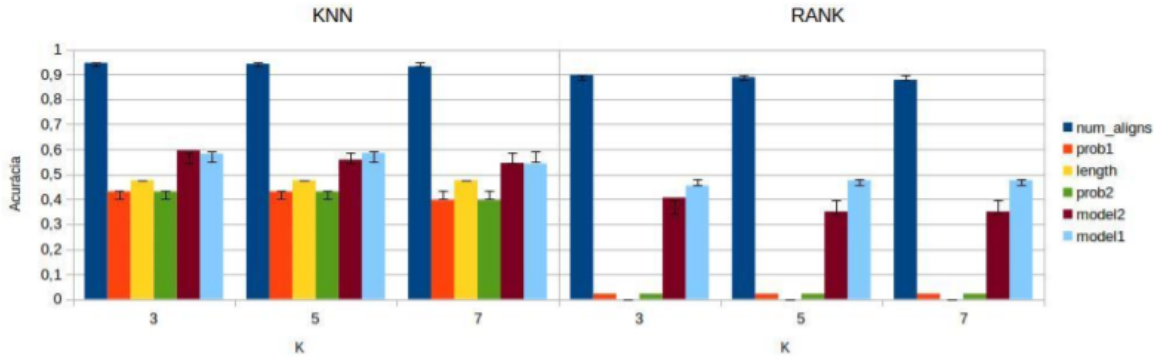


Figura 1. Accuracy Results for both Pure KNN and LabelRank Methods

Furthermore, the rank-based model obtained worse results than KNN. Although they both come close regarding the parameters num_aligns, $model\_1$, and $model\_2$, the rank accuracy results for the rest of the parameters are close to zero. One explanation for such behavior is the numerical nature of these attributes and their very sparse values in the input dataset, which produces very different rankings along the training.

5. Related Work

The ranking strategy used in this paper was also used in the recommendation context but applied to movies scenario [Pessiot et al. 2007]. The voting method has been used to decrease the training time with large datasets in a movies recommendation task, without deteriorating the quality of the recommendation [Das et al. 2014, Mukherjee et al. 2003]. Particularly, in [Mukherjee et al. 2003], Preference Learning is also used to leverage the recommendation. Regarding recommendation in workflows, Halioui [Halioui et al. 2016] combined natural language processing with ontologies to recommend searching keywords. In [Soomro et al. 2015], a pattern-based recommendation approach was built to suggest workflows composition. [Cheng et al. 2015] propose an approach for identifying and recommending the workflows for reference using semantic similarity. However, the aforementioned approaches do not recommend values to the parameters.

6. Conclusions and Future Work

The effective use of scientific workflows and SWfMS have fostered the scientific experimentation and its analysis. However, several experiments modeled as workflows have a large set of parameters to be configured, which is not a trivial task to accomplish. While in some cases the scientist may be working on an experiment where he/she already knows at least a subset of the most appropriate values, he/she may not know which is the best values to assign to the others parameters of the workflow. In addition, a poor choice of parameters may lead to undesired results and loss of time. In this paper, we propose a parameter recommendation algorithm called $FReeP$, based on Preference Learning and Voting Systems to recommend values for parameters, while restraining the recommendations to the user preferences. Experiments showed that when we employ a *KNN* classifier we can reach the correct parameter value in most cases, even in the presence of a reduced training set. On the other hand, we

still need to improve the methods based on *LabelRank* so that it can achieve its full potential. As future work, we plan: (i) to consider more clever partition strategies to make the training more efficient; (ii) to rely on a non-uniform choice from the KNN classifier; (iii) to explore other classifiers; and (iv) to test other voting schemas.

Referências

Black, D. (1976). Partial justification of the borda count. *Public Choice*, 28(1):1–15.

Cheng, Z., Zhou, Z., and Wang, X. (2015). Scientific workflow clustering and recommendation. In *11th International Conf. on Semantics, Knowledge and Grids (SKG)*, pages 272–274.

Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27.

Das, J., Mukherjee, P., Majumder, S., and Gupta, P. (2014). Clustering-based recommender system using principles of voting theory. In *IC3I*, pages 230–235. IEEE.

Fishburn, P. C. (1974). Simple voting systems and majority rule. *Systems Research and Behavioral Science*, 19(3):166–176.

Freire, J., Koop, D., Santos, E., and Silva, C. T. (2008). Provenance for Computational Tasks: A Survey. *Computing in Science & Engineering*, pages 20–30.

Fürnkranz, J. and Hüllermeier, E. (2011). Preference learning. In *Encyclopedia of Machine Learning*, pages 789–795. Springer.

Halioui, A., Valtchev, P., and Diallo, A. B. (2016). Towards an ontology-based recommender system for relevant bioinformatics workflows. *bioRxiv*, page 082776.

Herlocker, J. L., Konstan, J. A., Terveen, L. G., and Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53.

Hüllermeier, E., Fürnkranz, J., Cheng, W., and Brinker, K. (2008). Label ranking by learning pairwise preferences. *Artificial Intelligence*, 172(16-17):1897–1916.

Mukherjee, R., Sajja, N., and Sen, S. (2003). A movie recommendation system–an application of voting theory in user modeling. *User Modeling and User-Adapted Interaction*, 13(1-2):5–33.

Ocaña, K. A., de Oliveira, D., Ogasawara, E., Dávila, A. M., Lima, A. A., and Mattoso, M. (2011). Sciphy: a cloud-based workflow for phylogenetic analysis of drug targets in protozoan genomes. In *BSB11*, pages 66–70. Springer.

Pessiot, J., Truong, T., Usunier, N., Amini, M., and Gallinari, P. (2007). Learning to rank for collaborative filtering. In *ICEIS 2007 - Proc. of the 9th International Conf. on Enterprise Information Systems*, pages 145–151.

Refaeilzadeh, P., Tang, L., and Liu, H. (2016). Cross-validation. *Encyclopedia of database systems*, pages 1–7.

Soomro, K., Munir, K., and McClatchey, R. (2015). Incorporating semantics in pattern-based scientific workflow recommender systems: Improving the accuracy of recommendations. In *SAI'2015*, pages 565–571. IEEE.

Zhao, Y., Raicu, I., and Foster, I. (2008). Scientific workflow systems for 21st century, new bottle or new wine? In *IEEE Services*, pages 467–471. IEEE.

Time Series Forecasting for Purposes of Irrigation Management Process

Dieinison Braga[1], Ticiana L. Coelho da Silva[1], Atslands Rocha[1], Gustavo Coutinho[1], Regis P. Magalhães[1], Paulo T. Guerra[1], Jose A. F. de Macêdo[1]

[1]Federal University of Ceará (UFC)
Ceará – Brazil

dieinison@alu.ufc.br, {gustavolgcr, jose.macedo}@lia.ufc.br

{ticianalc, regismagalhaes, atslands, paulodetarso}@ufc.br

Abstract. *Irrigated agriculture is the most water-consuming sector in Brazil, representing one of the main challenges for the sustainable use of water. This study proposes and experimentally evaluates univariate time series models that predict the value of reference evapotranspiration, a metric of the water loss from crop to the environment. Reference evapotranspiration plays an essential role in irrigation management since it can be used to reduce the amount of water that will not be absorbed by the crop. The experiments performed under the meteorological dataset generated by a weather station. Moreover, the results show that the approach is a viable and lower cost solution for predicting ET_0, since only a variable needs to be monitored.*

1. Introduction

Population growth and changes in climate directly impact on worldwide food security. One of the primary objectives of agricultural research is to find improved ways to produce food. According to [Thiago et al. 2017], 72% of freshwater is consumed in irrigation, in Brazil. It is estimated that a massive portion of this amount is wasted due to poorly executed irrigation and lack of control from farmers about the exact amount of water to use in irrigation process.

Evapotranspiration value (ET_m) plays a key role in support to decision making in irrigation management, which is the simultaneous occurrence of evaporation and transpiration processes in a crop, measured in millimeters per a unit of time. We use the following equation to compute it: $ET_m = K_c \times ET_0$, where K_c is the crop coefficient c, given at INMET website[1], ET_0 is the reference crop evapotranspiration, which corresponds to the evapotranspiration rate of a grass surface. The value of ET_0 is very relevant to management and scaling in irrigation since it gives the information of how much water the crop loses to the environment [Thiago et al. 2017].

The traditional *Penman-Monteith* method [Allen et al. 1998] used to compute ET_0 is complex and does not tolerate the unavailability of some of its variables, which makes its use unfeasible. The paper [Caminha et al. 2017] proposes a Machine Learning-based approach to forecast ET_0 based on Linear Regression [James et al. 2013] and M5P [Wang and Witten 1996]. Despite the good results obtained in both techniques, they are

[1]http://sisdagro.inmet.gov.br/sisdagro/app/monitoramento/bhc

multivariate models, which means that it requires a weather station with many sensors to capture all the required variables, and there is no guarantee that models will fit, as well as in the absence of some variables.

Experiments performed by [Siami-Namini and Namin 2018] with univariate time series model demonstrated the Autoregressive Integrated Moving Average (ARIMA) [Box et al. 2015] model as a promising technique to achieve good accuracy performance in the forecast of financial time series. ARIMA model aims at describing the correlations in the data with each other. An improvement over ARIMA is Seasonal ARIMA (SARIMA) [Box et al. 2015], which takes into account the seasonality of dataset and was successfully used in short-term forecast [Tseng and Tzeng 2002]. In this paper, we use both approaches in our experiments.

The key contributions of this paper are: (i) offer an accurate and lower cost solution to estimate ET_0, since only a variable needs to be monitored; (ii) compare the performance of ARIMA, SARIMA, Linear Regression and M5P with respect to minimization achieved in the error rates in prediction; and (iii) release the dataset used in this work, for research and possible improvements by the scientific community.

The remaining sections of this article are organized as follows. Section 2 explains our proposed approach. Section 3 shows the steps necessary to accomplish our goals. Section 4 presents our experiments and its analysis. Finally, Section 5 summarizes this work and proposes future developments.

2. Time Series Forecasting

A time series (*TS*) is a series of data records indexed by dates. A time series model supposes that a series Z_t could be defined as $Z_t = T_t + S_t + \alpha_t$, being T the tendency, S the seasonality and α the white noise, at a moment t [Brockwell and Davis 2016]. Most of the *TS* models work on the assumption that the TS is stationary, i.e., its statistical properties such as mean and standard deviation remain constant over time. Due to many real-time series being non-stationary, statisticians had figured out ways to make *TS* stationary [Box et al. 2015].

In particular, differencing operator (∇) is a simple and efficient operator to transform a non-stationary TS to stationary. It is defined by the equation: $\nabla Z_t = Z_t - Z_{t-1}$, where Z is a TS at a moment t [Brockwell and Davis 2016]. In other words, we take the difference of the observation at a particular instant t with that at the previous instant $t-1$.

The ARIMA model takes three hyper-parameters p, d, q, which capture the key elements of the model, which are: (i) Autoregression (*AR*), a regression model that uses the relationship between an observation and a number (p) of lagged observations; (ii) Integrated (*I*), the number (d) of differentiation required to obtain stationarity; (iii) Moving Average (*MA*), an approach that takes into accounts the dependency between observations and the residual error terms when a moving average model is used for the lagged observations (q) [Box et al. 2015, Tseng and Tzeng 2002].

The SARIMA model incorporates both seasonal and non-seasonal factor in a TS data, its signature is $SARIMA(p, d, q) \times (P, D, Q)S$, where p and P are the non-seasonal and seasonal AR order; d and D are the non-seasonal and seasonal differencing; q and Q are the non-seasonal and seasonal MA order; and S is the time span of repeating seasonal

pattern, respectively [Tseng and Tzeng 2002].

3. Methodology

3.1. Data Collection and Cleaning

The climatic data were collected by a weather station, in the period from January, 1st to November, 29th of 2017 in the city of Quixadá, Ceará, Brazil. The original dataset contains 7941 hourly records, and it is composed of the features described in Table 1. This dataset is available in `https://github.com/Dieinison/ProjectET0/blob/master/dataset.csv`.

Table 1. Samples from dataset

Date	Atmospheric pressure		Air temperature			Relative humidity			Solar radiation		Temperature		Preciptation	Wind Speed	ET_0
	Max.	Min.	Max.	Min.	Mean	Max.	Min.	Mean	Total	Mean	Max.	Min.			
2017-11-29	620.5	599.7	21.4	19.6	32	55.2	45.3	50.1	1610	12.7	21.4	19.6	0.0	1.58	0.095
2017-11-29	620.2	599.7	21.7	19.4	32	52.3	41.9	46.9	1638	11.9	21.7	19.4	0.0	1.73	0.109
2017-11-29	620.4	599.6	20.9	19.1	34	45.8	39.7	42.3	1620	19	20.9	19.1	0.0	2.10	0.147
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

We aggregated the original hourly data on a daily basis. Furthermore, we detected outliers observations through *Proximity-Based Outlier Detection* technique [Tan et al. 2006] and remove them. The tuples contain the values described in Table 2 were removed. At the end of this procedure, 333 tuples remained.

Table 2. Removed instances

Precipitation ≥ 60
Minimum temperature ≤ 0
Minimum relative humidity ≤ 20

3.2. Prediction models

To create the prediction models, we split the dataset into 80% for training and 20% for testing. Each algorithm produced its particular model using the attributes taken as input. Thus, we generated four distinct models, Linear Regression and M5P were created from all the attributes of the dataset, ARIMA and SARIMA models were generated only with ET_0. These models and their comparisons are presented in Section 4.

For purposes of comparisons between the models generated, we used the same dataset (given by weather station from UFC Quixadá). We performed the prediction models by applying the Linear Regression and M5P algorithms, both implemented in the WEKA[2] tool.

In order to forecast through ARIMA and SARIMA, we perform the Box-Jenkins methodology [Box et al. 2015], defined as: (i) identification of the model, i.e., finding the appropriate orders for p, d, q, P, D, Q, S; (ii) estimation of the unknown parameters; (iii) validation of the model; and (iv) forecast future outcomes based on the known data.

[2]https://www.cs.waikato.ac.nz/ml/weka/

3.3. Models Evaluations

To evaluate both techniques, the Root Mean Squared Error(RMSE) and Mean Absolute Error (MAE) are calculated as the evaluation metrics of the performance, defined by

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \quad , \quad MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

where i is the sample index, n is the total number of observations, y is the expected attribute value and $\hat{y}$ is the value output by the algorithm used [James et al. 2013]. Both metrics can range from 0 to ∞. They are negatively-oriented scores, which means lower values are better. RMSE has the benefit of penalizing large errors, while MAE is a measure of average error.

4. Experiments and Results

As stated earlier, these experiments used a real dataset with observations collected from a weather station located in Campus UFC Quixadá, in Brazil.

Initially, we generated the Machine Learning-based approaches, through WEKA tool. Due to lack of space, we do not present in this paper our Linear Regression and M5P prediction models. They are available in `http://bit.ly/result_linear_regression` and `http://bit.ly/result_m5p`, respectively.

With the view to generate time series models, we checked stationarity by plotting rolling average and rolling standard deviation as shown in Fig.1. The evaluated mean and standard deviation show significant instability over time, suggesting the data is non-stationary. Another technique to evaluate the non-stationary is the Dickey-Fuller (DF) test. The DF is a unit root test that evaluates the strength of trend in a time series component [P. Avishek 2017]. The output for DF test is shown in Table 3. As we can see, the DF Statistic is higher than the critical values, so this series is non-stationary. Therefore we can approach this with ARIMA models.

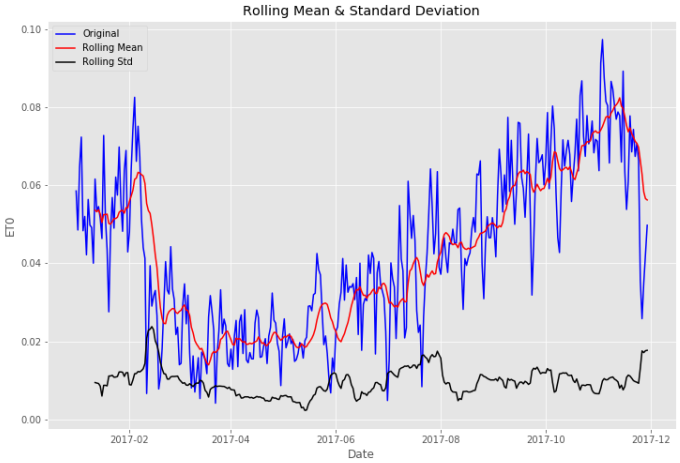


Table 3. Results of DF Test

DF Statistic	-1.695411
Critical Value 1%	-3.450695
Critical Value 5%	-2.870502
Critical Value 10%	-2.571545

Figure 1. Original ET_0.

In order to obtain the optimal hyper-parameters for ARIMA and SARIMA models, we used a function, called *auto arima*, from Pyramid[3], an API under MIT License

[3] `https://github.com/tgsmith61591/pyramid`

that provide an systematic approach to find the best hyper-parameters, based on a given information criteria, which in this case will be the Corrected Akaike Information Criterion (AIC_c), as recommended in [Brockwell and Davis 2016]. This criterion includes a penalty term to discourage the fitting of too many parameters, i.e., the fitted model with the smaller value of AIC_c will be the best choice [Smith 2017, P. Avishek 2017]. Tables 4 and 5 present the parameters output by *auto arima* function for ARIMA and SARIMA models, respectively.

Table 4. ARIMA parameters.

Parameter	Value
AR order p	1
Difference order d	1
MA order q	1

Table 5. SARIMA parameters.

Parameter	Value
AR order p	1
Difference order d	1
MA order q	1
Seasonal AR order P	0
Seasonal difference D	1
Seasonal MA order Q	2
S	12

Table 6 shows RMSEs and MAEs generated from models. As we can see, the univariate ARIMA and SARIMA models presented error values very low as it is close to zero. A value of RMSE or MAE equals to zero would that the estimator is predicting observations with perfect accuracy. Besides, in Table 7, we showed statistical properties of our label variable, ET_0, thus, as errors rates (RMSE and MAE) are less than the standard deviation, our results indeed show a good accuracy [Legates and McCabe 1999].

The results show an outperformance of multivariate model M5P, under RMSE and MAE metrics, over univariate time series models. Nevertheless, univariate time series models show us that these models indeed fit well the data, since there were small differences between predictions and expected values. Regarding *TS* models, ARIMA outperformed SARIMA in both metrics, indicating that our data is better fitted by a non-seasonal model.

Table 6. Metrics comparisons between techniques.

Model	RMSE	MAE
ARIMA	0.0196	0.0173
Linear Regression	0.0072	0.0056
M5P	0.0070	0.0056
SARIMA	0.0225	0.0201

Table 7. Mean and Standard Deviation of observed ET_0.

Statistic	Value
Mean	0.0430
Standard deviation	0.0462

Due to the costs of owning a weather station with many sensors, capture all the variables required for multivariate models might not be affordable for low-income farmers. In contrast, the results show us that an ARIMA model is an affordable solution for predicting ET_0 since only a variable needs to be monitored, with no need of multiples sensors.

5. Conclusion

This paper compares the accuracy of univariate ARIMA and SARIMA models with multivariate Machine Learning-based algorithms, Linear Regression and M5P. The results show that M5P outperform the other techniques. Despite that, this paper advocates the benefits of applying univariate time series algorithms to predict ET_0, since these models presented small differences between predictions and expected values, i.e., good accuracy. Besides, *TS* models might be an affordable solution for low-income farmers, since only a variable needs to be monitored. For future works, we aim at improving and validating our proposed models for other datasets and compare with deep learning based approaches.

Acknowledgment

The authors acknowledge FUNCAP for the research supported.

References

Allen, R. G., Pereira, L. S., Raes, D., Smith, M., et al. (1998). Crop evapotranspiration-guidelines for computing crop water requirements-fao irrigation and drainage paper 56. *FAO, Rome*, 300(9):D05109.

Box, G. E., Jenkins, G. M., Reinsel, G. C., and Ljung, G. M. (2015). *Time series analysis: forecasting and control*. John Wiley & Sons, New Jersey, USA.

Brockwell, P. J. and Davis, R. A. (2016). *Introduction to time series and forecasting*. Springer, Switzerland.

Caminha, H., Silva, T., Rocha, A., and Lima, S. (2017). Estimating reference evapotranspiration using data mining prediction models and feature selection. *Proceedings of the 19th International Conference on Enterprise Information Systems (ICEIS 2017)*, 1:272–279.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An introduction to statistical learning*, volume 112. Springer, New York, USA.

Legates, D. R. and McCabe, G. J. (1999). Evaluating the use of goodness-of-fit measures in hydrologic and hydroclimatic model validation. *Water resources research*, 35(1):233–241.

P. Avishek, P. P. (2017). *Practical Time Series Analysis: Master Time Series Data Processing, Visualization, and Modeling using Python*. Packt, Birmingham, UK.

Siami-Namini, S. and Namin, A. S. (2018). Forecasting economics and financial time series: Arima vs. lstm. *arXiv preprint arXiv:1803.06386*.

Smith, T. G. (2017). *Pyramid: ARIMA estimators for Python*. MIT, USA.

Tan, P.-N. et al. (2006). *Introduction to data mining*. Pearson Education India, India.

Thiago, H. F., Wagner, M. d. C., and Marcus, A. F. (2017). *Atlas irrigação: uso da água na agricultura irrigada*. Agência Nacional de Águas, Brasília, DF, Brasil.

Tseng, F.-M. and Tzeng, G.-H. (2002). A fuzzy seasonal arima model for forecasting. *Fuzzy Sets and Systems*, 126(3):367–376.

Wang, Y. and Witten, I. H. (1996). Induction of model trees for predicting continuous classes.

Uma Abordagem para Caracterização de documentos RDF através de Esquemas Conceituais

Alisson S. Maia[1], **Vagner Pagotti**[1], **Rebeca Schroeder**[1]

[1]Departamento de Ciências da Computação
Universidade do Estado de Santa Catarina (UDESC)
Centro de Ciências Tecnológicas – 89.219-710 – Joinville – SC – Brasil

{alisson.maia11,pagotti}@gmail.com, rebeca.schroeder@udesc.br

Abstract. *A suitable storage model for RDF depends on a set of data characteristics and the knowledgment of its schema. This paper aims to contribute in this context on providing a method to extract conceptual schemas from RDF documents. The goal is to characterize an RDF data structure through an entity-relationship schema and its constructors. The proposed method is evaluated by a case study which demonstrates that the conceptual schemas generated are valid according to the model proposed by a benchmark for RDF.*

Resumo. *Dentre as possibilidades de bancos de dados para o armazenamento de dados RDF, a escolha de um modelo adequado depende de um conjunto de características dos dados e a compreensão de seu esquema. Este trabalho visa contribuir para este contexto através de um método de extração de esquemas conceituais a partir de documentos RDF. O objetivo deste método é caracterizar a estrutura de dados RDF através da produção de um esquema entidade-relacionamento e seus construtores. O método proposto foi avaliado por um estudo de caso que demonstrou que os esquemas conceituais gerados são válidos de acordo com o modelo proposto por um benchmark para RDF.*

1. Introdução

Como um modelo em evidência no contexto da Web Semântica, RDF se tornou alvo de uma série de trabalhos que propõem metodologias para armazenar seus documentos em diferentes tipos de Sistemas de Banco de Dados. Grande parte destes trabalhos apresentam suas propostas de armazenamento RDF para o modelo relacional [Scabora et al. 2017]. Uma vez que dados RDF são descritos no formato de triplas contendo sujeito, predicado e objeto, o mapeamento direto para o relacional corresponde à construção de uma única tabela contendo estes 3 campos. Bancos de dados relacionais que adotam esta estratégia são conhecidos como *triple-stores* [Neumann and Weikum 2010]. Porém, esse método de armazenamento não possui bom desempenho, uma vez que consultas nessa tabela implicam na execução de auto-junções para recuperar triplas relacionadas [Zeng et al. 2013]. Neste caso, fica evidente que compreender a estrutura de dados RDF, e como eles se relacionam, pode favorecer à construção de esquemas de bancos de dados mais apropriados. Embora RDF seja qualificado como um modelo livre de esquema, os autores de [Pham et al. 2015] identificaram que é possível extrair a estrutura de dados RDF para um grande número de *datasets* deste modelo.

Como uma alternativa aos *triple-stores*, alguns trabalhos propõem observar a estrutura dos dados RDF para criar um esquema relacional capaz de agrupar em tabelas

os atributos de uma mesma classe de dados [Ramanujam et al. 2009] [Pham et al. 2015]. Entretanto, a noção das estruturas extraídas são diretamente representadas no modelo relacional, o que pode limitar a representação destes dados através de outros modelos de banco de dados. Neste sentido, este trabalho propõe extrair a estrutura de dados RDF e representá-la através de um esquema conceitual definido através do modelo entidade-relacionamento (ER). Além de servir como base para o projeto de qualquer modelo de banco de dados, a abstração fornecida por esquemas conceituais pode contribuir como uma visão unificada dos conceitos de um domínio relacionado a um conjunto de dados RDF.

A proposta deste trabalho também se diferencia dos trabalhos de [Ramanujam et al. 2009] e [Pham et al. 2015] por extrair algumas métricas que caracterizam a variabilidade da estrutura de *datasets* RDF. Estas métricas dizem respeito às cardinalidades de relacionamentos e atributos do esquema conceitual. A extração destas métricas foi inspirada pelo trabalho de [Duan et al. 2011], em que informações similares foram extraídos diretamente de documentos RDF. Diferente deste trabalho, este artigo apresenta métricas equivalentes mas sobre um esquema conceitual. Acredita-se que tal conhecimento sobre a estruturabilidade dos dados em nível conceitual possa embasar decisões em um futuro projeto de banco de dados, desde a escolha pelo modelo de dados mais apropriado, até o esquema lógico mais adequado.

Este artigo está organizado em mais 3 seções. A Seção 2 apresenta o método de extração de esquemas conceituais a partir de documentos RDF. A seção seguinte apresenta um estudo de caso utilizando a metodologia proposta sobre um gerador de *datasets* de um *benchmark* para RDF. As conclusões deste trabalho são apresentadas pela Seção 4, em conjunto com os trabalhos futuros.

2. Caracterização de Documentos RDF

Dados RDF são representados por triplas definidas por expressões do tipo `sujeito-propriedade-objeto (s, p, o)`. Um documento RDF é constituído por um conjunto de triplas, e pode ser representado por um grafo direcionado, onde os vértices são sujeitos e objetos, e as arestas correspondem às propriedades que os interliga. Um exemplo deste tipo de grafo é apresentado na parte direita da Figura 1. A aresta rotulada com `feature` que conecta os vértices `Produtct1` e `ProductFeature1` representa, por exemplo, a tripla `Produtct1-feature-ProductFeature1`.

Embora RDF corresponda a um modelo livre de esquema, é possível extrair tipos de dados de seus próprios documentos assim como apontado em [Duan et al. 2011]. Em um documento, triplas com a propriedade `http://www.w3.org/1999/02/22-rdf-syntax-ns#type` (`type`) determinam o tipo de seus respectivos sujeitos. A Figura 1 ilustra a extração destes tipos. Por exemplo, a tripla `Produtct2-type-Product` determina que o sujeito `Product2` é do tipo `Product`. Na Figura, os tipos são diretamente representados por entidades do modelo ER. Observe que, a partir da identificação de tipos, é possível identificar como as instâncias de um tipo são estruturadas em termos de atributos e relacionamentos com instâncias de outros tipos. Para tanto, a abordagem proposta por este artigo se baseia neste tipo de análise para sumarizar a estrutura de instâncias de tipos RDF em um esquema conceitual definido no modelo ER.

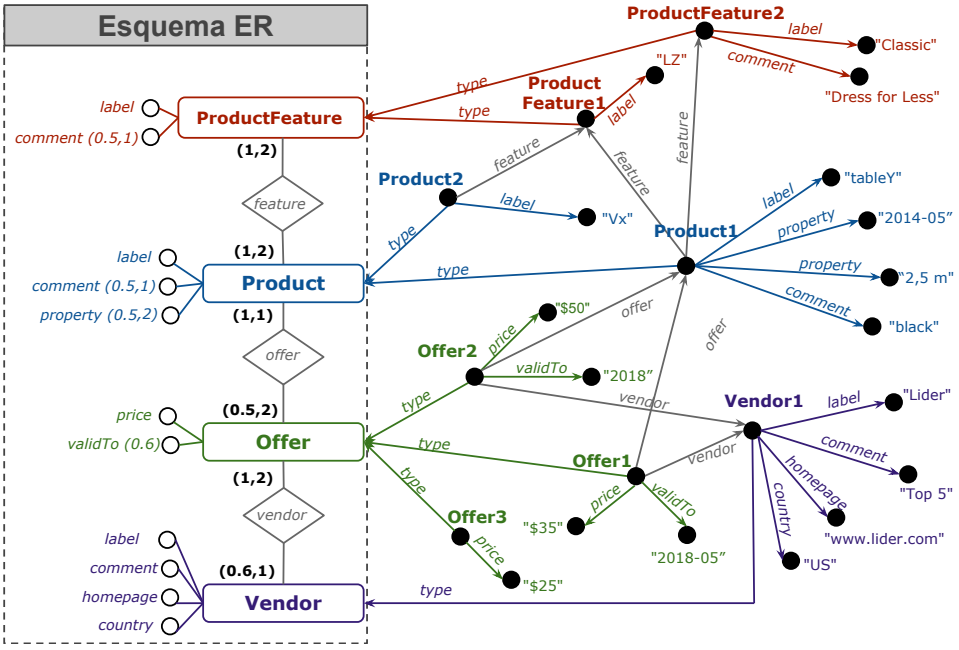


Figura 1. Extração de Esquemas a partir de triplas RDF

Nas seções a seguir são apresentadas as definições que determinam a extração de entidades, atributos e relacionamentos em um esquema ER. Além disto, as cardinalidades de atributos e relacionamentos são apresentadas na sequência.

2.1. Extração de Entidades, Atributos e Relacionamentos

Um esquema conceitual ER S é definido como $S = \{E, R\}$, onde E é o conjunto de entidades e R o conjunto de relacionamentos que representam associações entre duas entidades. Assim como ilustrado pela Figura 1, o conjunto de entidades pode ser extraído de um documento RDF a partir de triplas contendo a propriedade `type`. Logo, a partir de um documento RDF D, o conjunto de entidades é obtido por $E = \{e | \exists(s, \texttt{type}, e) \in D\}$.

No exemplo, E corresponde a $\{\texttt{ProductFeature}, \texttt{Product}, \texttt{Offer}, \texttt{Vendor}\}$. Assim, o conjunto de instâncias de uma entidade $e \in E$ pode ser definido como $I(e) = \{s | \exists(s, \texttt{type}, e) \in D\}$. Por exemplo, $I(\texttt{Product}) = \{Product1, Product2\}$. A partir deste ponto, será utilizado $I(E)$ para representar o conjunto de instâncias de todas as entidades em E. No exemplo, $I(E) = \{\texttt{Product1}, \texttt{Product2}, \texttt{ProductFeature1}, \texttt{ProductFeature2}, \texttt{Offer1}, \texttt{Offer2}, \texttt{Vendor1}\}$.

Os atributos de entidades correspondem a propriedades em D que associam instâncias de E com valores literais. O conjunto de atributos de uma entidade $e \in E$ é definido por $A(e) = \{p | s \in I(e), o \notin I(E), \exists(s, p, o) \in D\}$. Logo, considera-se como valores literais os objetos que não correspondam a instâncias de E. No exemplo, os atributos de `Product` são definidos como $A(\texttt{Product}) = \{\texttt{label}, \texttt{comment}, \texttt{property}\}$. Embora não apresentado no exemplo, considera-se para todas as entidades que um atributo identificador será automaticamente criado, cujo valor conterá a URI de sujeitos que correspondem a elementos de $I(E)$. Além disto, atributos obrigatórios e opcionais serão identificados pela extração de suas respectivas cardinalidades a ser apresentado na

próxima seção.

Os relacionamentos são extraídos a partir das propriedades que associam duas instâncias diferentes de E. Logo, o conjunto de relacionamentos de S é definido por $R = \{r|s \in I(E), o \in I(E), \exists(s,r,o) \in D\}$. De acordo com o exemplo, $R = \{\texttt{feature}, \texttt{offer}, \texttt{vendor}\}$. Dado que D é um grafo direcionado, denomina-se $\texttt{out}(r)$ a entidade que corresponde à entidade origem da direção apontada pelo relacionamento, assim como $\texttt{in}(r)$ a entidade destino. No exemplo, $\texttt{out}(\texttt{feature})$ corresponde à entidade `Product` e $\texttt{in}(\texttt{feature})$ à entidade `ProductFeature`. Uma vez que os relacionamentos são extraídos diretamente pelas associações estabelecidas entre sujeitos e objetos de triplas RDF, não é possível a extração de relacionamentos n-ários e de atributos para relacionamentos. Entretanto, considera-se que ambas construções de um modelo ER possam ser respectivamente representadas por um conjunto de relacionamentos binários e atributos de entidades relacionadas. Ademais, o método de extração das cardinalidades mínimas e máximas dos relacionamentos é apresentado pela próxima seção.

2.2. Extração de Cardinalidades

Como apresentado pela Seção 2.1, tanto os atributos quanto os relacionamentos são extraídos a partir de propriedades das triplas de D. Da mesma forma, conhecer a frequência com que estas propriedades ocorrem para instâncias de entidade determina a cardinalidade de atributos e relacionamentos associados.

Para atributos de entidade, a cardinalidade mínima define a quantidade mínima de valores associados a cada instância de uma entidade, qualificando atributos opcionais ou obrigatórios. A cardinalidade mínima para um atributo a de uma entidade e é definida por:

$$min\_card(a) = \frac{\mid \{s|s \in I(e),\ a \in A(e),\ \exists(s,a,o) \in D\} \mid}{\mid I(e) \mid} \tag{1}$$

Observe que a Equação 4 não tem por objetivo contabilizar a quantidade de ocorrências de uma dada propriedade sobre todas as instâncias de e, e sim a quantidade de instâncias que possuem tal propriedade associada. Assim sendo, dadas as duas instâncias de `Product` do exemplo, o atributo `label` é qualificado como obrigatório, isto é $\texttt{min\_card(label)}=1$, visto que as duas instâncias de `Product` contêm esta propriedade. Entretanto, para o atributo `comment` a cardinalidade mínima obtida é 0.5, uma vez que apenas `Product1` contém esta propriedade. Logo, `comment` corresponde a um atributo opcional. Desta forma, a cardinalidade mínima de um atributo pode assumir valores entre 0 e 1 (inclusive). Por se tratar de um modelo semi-estruturado, valores de cardinalidade menores que 1 indicam não somente que o atributo é opcional, mas também a proporção de instâncias de entidade que apresentam valores para esta propriedade.

A cardinalidade máxima de um atributo é extraída com base na ocorrência máxima encontrada em uma das instâncias de sua respectiva entidade. Para tanto, dado o total de instâncias de uma entidade e, a cardinalidade máxima de um atributo a desta entidade é definida por uma instância e_i que maximiza a seguinte expressão:

$$max\_card(a) = max(\mid \{a|e_i \in I(e),\ a \in A(e),\ \exists(e_i,a,o) \in D\} \mid) \tag{2}$$

De acordo com o exemplo da Figura 1, a cardinalidade máxima do atributo `property` em `Product` é igual a 2. Quanto aos demais atributos, a ocorrência máxima

é igual a 1. Para efeito de simplificação do exemplo, atributos que não apresentam cardinalidades mínimas e máximas anotadas correspondem a (1,1), isto é, atributos obrigatórios e mono-valorados.

De forma análoga aos atributos, as cardinalidades dos relacionamentos são obtidas pela avaliação das propriedades associadas. Entretanto, neste caso, as cardinalidades são dadas para cada uma das entidades que participam de um relacionamento. Logo, dado um relacionamento r definido entre duas entidades, a cardinalidade mínima de cada uma das entidades participantes é definida por:

$$min\_card(e, r) = \frac{\mid \{e | e \in I(E),\ r \in R,\ (\exists(e, r, o) \in D\ ou\ \exists(s, r, e) \in D)\} \mid}{\mid I(e) \mid} \quad (3)$$

Em virtude da direção das arestas em D, a Equação 6 considera que as arestas relacionadas a r são dadas em D em apenas uma direção. Desta forma, se $\texttt{out}(\texttt{r}) = e$ haverá uma tripla $(e, r, o) \in D$ para uma das instâncias de e. Caso contrário, se $\texttt{in}(\texttt{r}) = e$, haverá uma tripla $(s, r, e) \in D$. Como exemplo, a cardinalidade mínima da entidade `Offer` no relacionamento `vendor` é igual a 0.6, uma vez que das 3 instâncias de `Offer`, apenas 2 estão relacionadas por `vendor`. Entretanto, a cardinalidade mínima de `Vendor` no relacionamento `vendor` é igual a 1 em virtude de que a única instância desta entidade está associada pelo relacionamento.

A cardinalidade máxima de uma entidade e em um relacionamento r é extraída com base na ocorrência máxima encontrada em uma das instâncias de suas entidades. Para tanto, dado o total de instâncias de uma entidade e, sua cardinalidade máxima em r é definida por uma instância e_i que maximiza a seguinte expressão:

$$max\_card(e, r) = max(\mid \{a | e_i \in I(e),\ (\exists(e_i, r, o) \in D\ ou\ \exists(s, r, e_i) \in D)\} \mid) \quad (4)$$

No exemplo da Figura 1, $\texttt{max\_card}(\texttt{Product}, \texttt{offer}) = 2$ cujo valor máximo é determinado pela instância `Product1`. Já $\texttt{max\_card}(\texttt{Offer}, \texttt{offer}) = 1$, visto que todas as instâncias de `Offer` estão associadas a no máximo 1 instância de `Product`.

A extração de entidades, atributos e relacionamentos pode ser derivada de abordagens que fazem o mapeamento do modelo RDF para o relacional. Entretanto, em virtude da fraca abstração do modelo relacional, a noção de cardinalidades mínimas e máximas não pode ser identificada. Considera-se que esta noção é fundamental para, por exemplo, evitar a geração de tabelas com muitos atributos opcionais.

3. Estudo de Caso

Um protótipo que implementa o método de extração proposto foi desenvolvido para avaliação dos esquemas produzidos. Para a estudo, foi utilizado o gerador de documentos RDF fornecido pelo *Berlin SPARQL Benchmark* (BSBM)[Bizer and Schultz 2009]. O BSBM é baseado em um caso de uso de um sistema de *e-commerce*, onde uma lista de produtos é oferecida por vendedores. Parte do esquema de dados do BSBM foi utilizado no exemplo da Figura 1. Para a geração das bases, o *benchmark* utiliza um fator de escala baseado no número de produtos a gerar. Neste estudo aplicou-se o fator de 100 produtos, produzindo um documento com 50 mil triplas RDF.

Um esquema ER é apresentado pela própria documentação do BSBM. Para tanto, comparou-se o esquema ER do BSBM com o produzido pelo protótipo do método aqui proposto. Observou-se que todas as entidades, atributos e relacionamentos foram extraídos adequadamente. No entanto, com relação às cardinalidades houve pequenas divergências relacionadas às cardinalidades mínimas de 3 entidades em seus relacionamentos. Neste caso, o documento RDF foi verificado e constatou-se que as cardinalidades não correspondiam às indicadas na documentação do BSBM, e sim às geradas pelo protótipo desenvolvido.

4. Conclusões

Este artigo propõe um método para a extração de esquemas conceituais ER a partir de documentos RDF. A solução apresentada difere de trabalhos relacionados por representar a estrutura de dados através de um modelo mais abstrato, se comparado ao modelo relacional utilizado por estes trabalhos. Os esquemas produzidos pelo método proposto podem ser utilizados para apoiar decisões do projeto de um BD para armazenamento de dados RDF. Por exemplo, o conhecimento da cardinalidade mínima de atributos pode evitar a criação de tabelas com excesso de colunas opcionais, caso um BD relacional venha a ser escolhido. Sobretudo, os esquemas produzidos são capazes de expressar o grau de estruturabilidade de dados RDF através da opcionalidade de relacionamentos e atributos indicado por suas respectivas cardinalidades.

Um estudo de caso foi realizado em que se verificou que os esquemas produzidos por este trabalho correspondem aos esquemas de um *benchmark* para RDF. Entretanto, experimentos com *datasets* reais e maiores se fazem necessários para a validação do método sobre fontes com um grau de estruturabilidade provavelmente menor. Além disto, considera-se como trabalho futuro a identificação de variações na estrutura dos dados em decorrência à evolução de esquemas RDF.

Referências

Bizer, C. and Schultz, A. (2009). The Berlin SPARQL Benchmark. *Int. J. Semantic Web Inf. Syst.*, 5(2):1–24.

Duan, S., Kementsietsidis, A., S., K., and U., O. (2011). Apples and oranges: A Comparison of RDF Benchmarks and Real RDF Datasets. In *SIGMOD*, pages 145–156.

Neumann, T. and Weikum, G. (2010). The RDF-3X Engine for Scalable Management of RDF Data. *The VLDB Journal*, 19(1):91–113.

Pham, M.-D., Passing, L., Erling, O., and Boncz, P. (2015). Deriving an Emergent Relational Schema from RDF Data. In *WWW*, pages 864–874.

Ramanujam, S., Gupta, A., Khan, L., Thuraisingham, B., and Seida, S. (2009). R2D: Extracting Relational Structure from RDF Stores. In *International Joint Conference on Web Intelligence and Intelligent Agent Technology*, pages 361–366.

Scabora, L. C., Oliveira, P. H., dos Santos Kaster, D., Traina, A. J. M., and Traina, C. (2017). Relational graph data management on the edge: Grouping vertices' neighborhood with Edge-k. In *Simpósio Brasileiro de Banco de Dados*, pages 124–135.

Zeng, K., Yang, J., Wang, H., Shao, B., and Wang, Z. (2013). A Distributed Graph Engine for Web Scale RDF Data. *Proceedings of the VLDB Endowment*, 6(4):265–276.

Pytology: Rumo ao Cálculo de Relevância sobre dados RDF

Victor V. Barros Leal[1], **José Antônio F de Macedo**[1], **Lucas Peres Gaspar**[1] **e David Araújo Abreu**[1]

[1]Insight Data Science Lab – Universidade Federal do Ceará (UFC)
Fortaleza – CE – Brazil

{victorbl,lucasperes,araujodavid}@lia.ufc.br, jose.macedo@dc.ufc.br

Abstract. *With the wide availability of RDF datasets on the Web, it becomes increasingly complex the manual analysis to understand the domains of ontologies and their levels of links. Therefore, a challenge is the semi-automatic identification of the relevant relations at the ontology, which are important to define the semantics of the data. This work presents a method to calculate relevance values to the predicates in an ontology by using topological analysis. We show the consolidation of this work with a tool named Pytology and the experimental results generated by using available datasets on the web.*

Resumo. *Com a ampla disponibilidade de bases RDF na Web, torna-se cada vez mais complexa a análise manual para entendimento dos domínios das ontologias e seus níveis de ligações. Diante disso, um desafio é a identificação semi-automática das relações relevantes na ontologia, as quais sejam importantes para definir a semântica dos dados. Este trabalho apresenta um método para calcular valores de relevância para os predicados de uma ontologia através de métricas de análise topológica. Apresentamos a consolidação do trabalho na ferramenta Pytology e nos resultados de experimentos em bases disponíveis na web.*

1. Introdução

Fontes de dados RDF têm ganhado grande importância, aumentando o número de ferramentas para manipulá-los. [Crubézy and Musen 2004] demonstra como o uso dessas fontes é importante para a solução de problemas em diversos cenários de integração de dados. No entanto, a grande quantidade de fontes RDF cria um desafio para os usuários que têm que realizar busca ou *surfing* sobre esses dados, visto que muitos usuários não possuem conhecimento prévio sobre o conteúdo de tais fontes. Esses desafios são trabalhados em áreas como Busca Exploratória [Marchionini 2006] e *Information Retrieval* [Auer et al. 2007].

O problema de recuperar informações de bases em RDF a partir de uma busca é abordado em vários artigos como mostra [Roa-Valverde and Sicilia 2014]. [Elbassuoni and Blanco 2011] apresenta a utilização de cálculos estatísticos pra identificar dados relevantes a partir de palavras-chaves, entretanto definir a relevância aos predicados é um problema nessa abordagem. No contexto de busca exploratória esse problema se apresenta de outra maneira e com poucas soluções, conforme destacado em [Mirizzi and Di Noia 2010] e [Musetti et al. 2012], onde, a partir de um termo do grafo, navega-se sobre os dados a partir dos predicados. Nesses trabalhos é possível perceber a necessidade de calcular a relevância dos predicados.

A eficiência das abordagens citadas depende de um valor de *relevância* para os predicados, os quais não são definidos automaticamente. Para contornar esse problema, os trabalhos definem, manualmente, um conjunto de valores ou regras para cada domínio de dados ou utilizam glossários com predicados mais relevantes. A relevância nesse contexto é uma medida, baseada em alguma métrica, que permite comparar os diferentes tipos de predicados.

Neste trabalho, apresentamos um método que permite calcular valores de relevância dos predicados de qualquer base RDF utilizando-se da análise topológica dos dados. A consolidação desse trabalho dá-se pelo Pytology, uma ferramenta que implementa esse método e gera valores de relevância a partir de medidas de centralidades de grafos. Destarte, como relevância é um conceito subjetivo, os valores da ferramenta servem como uma orientação baseadas em medidas já conhecidas. O resto desse trabalho está dividido da seguinte forma: na seção 2 explicaremos a ferramenta, expondo uma visão geral e como o método funciona. Na seção 3 apresentaremos alguns experimentos e, por fim, traremos nossas conclusões e trabalhos futuros na seção 4.

2. Pytology

2.1. Visão Geral

Dentre as técnicas de cálculo de centralidade sobre grafos, é comum o cálculo de valores apenas para os nós. Isso ocorre pois, geralmente, as arestas não carregam uma informação própria que as caracterize (além, claro, dos nós que elas conectam). Diante disso, é proposto um método que, a partir do cálculo de centralidade dos nós de um grafo RDF, calcula-se um valor que representa a relevância das relações. A Figura 1 demonstra a organização do Pytology.

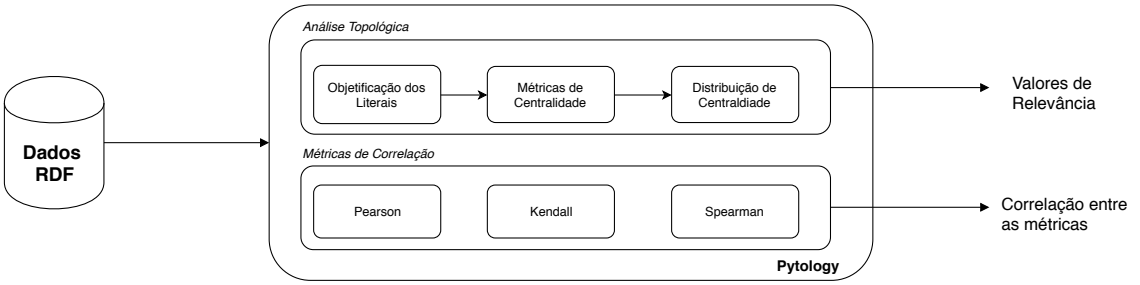


Figura 1. Visão geral das componentes do Pytology

O processo inicia a partir das instâncias de dados os quais são representadas como um grafo RDF que em particular pode conter uma ontologia. Em um grafo RDF os predicados compõem as arestas e os termos e literais os vértices. Em seguida, aplica-se um pré-processamento definido como **objetificação de literais**. Utiliza-se essa etapa para melhorar a representação das relações que apontam para literais. Em seguida, calcula-se a relevância dos nós do grafo a partir de algoritmos de centralidade em grafos. Por fim, aplica-se um cálculo para distribuir esse valores sobre as arestas.

2.2. Análise Topológica

A análise topológica é a parte principal do Pytology. Ela ocorre em três etapas: **objetificação de literais**, **cálculo de centralidade** e **distribuição de centralidade**.

2.2.1. Objetificação dos Literais

Em bases RDF, existem dois tipos de propriedades: relacionamentos, que relacionam duas instâncias de dados (classes) e atributos, que relacionam instâncias à literais (números, datas, texto, etc.). Em um grafo RDF, caso duas instâncias de dados apontem para literais iguais, como uma mesma data, por exemplo, o grafo reconheceria essas datas como dois nós diferentes. Por exemplo: suponha duas entidades com mesmo idioma (dado1, idioma, ”pt”), (dado2, idioma, ”pt”). O relacionamento ”idioma”estaria mais disperso nesse grafo, além de que haveria dois nós distintos para representar o literal ”pt”.

O processo de objetificação transforma esses literais em instâncias de dados. Isso permite que, caso dois ou mais literais sejam iguais em seu valor, eles sejam representados como um único nó no grafo, relacionando-se a todos os dados que apontam para aquele literal. Partindo do exemplo do idioma acima: cria-se um novo dado para intermediar essa relação, gerando as seguintes relações: (dado1, idioma, idioma1), (dado2, idioma, idioma1), (idioma1, valor, ”pt”). Observa-se que a informação do idioma de dado1 e dado2 não é perdida, agora apontam para um mesmo nó, o qual representa o idioma ”pt”.

2.2.2. Métricas de Centralidade

Como os dados estão representados em um grafo RDF, podemos utilizar técnicas de *network-analysis* [Freeman 1978] sobre os dados para realizar cálculos de centralidade dos nós.

Atualmente, o Pytology trabalha com as métricas $Betweenness$, $Closeness$, $Eigenvectors$, $Katz-Eigenvector$ e $Pagerank$. Com exceção do $Betweenness$, essas métricas não calculam valores para as arestas, apenas para os nós. Logo, faz-se necessário a etapa de distribuição de centralidade para podermos mensurar a valor das arestas.

É importante mencionar que cada técnica tem uma semântica associada ao seu resultado. O $Closeness$, por exemplo, dará maior importância a termos mais centrais do grafo, enquanto o $Betweenness$ dará mais importância a entidades limiares entre grupos. Logo, a semântica associada ao valor de relevância dos predicados dependerá da métrica de centralidade utilizada.

2.2.3. Distribuição da centralidade para as arestas

Em um grafo RDF, uma mesma relação pode ocorrer entre diversos pares de dados. Como existem múltiplas arestas no grafo que representam o mesmo predicado, não se pode apenas atribuir o valor de centralidade dos nós que elas conectam como seu próprio valor de relevância. Para calcular esse valor, é preciso levar em conta todas as suas ocorrências.

Se um certo predicado R ocorre entre nós de alta relevância, é provável que essa aresta deva ser de grande relevância. Analogamente, se ela ocorre entre nós de baixa relevância, ela deva ser de baixa relevância. Porém, esse predicado pode ocorrer entre nós de altas e baixas relevâncias, além de poder aparecer mais de uma vez a partir de um mesmo nó. Portanto, para distribuir as relevâncias para as arestas, realiza-se uma média ponderada com a ocorrência da relação e a relevância do nó a quem ela se refere, através

da seguinte fórmula:

$$C_p = \frac{\sum_n^{n \in G} C_n * F_p^n}{F_p}$$

A relevância C_p de um certo predicado r é calculada somando, para cada nó n do grafo G, o produto entre a relevância C_n do nó n pelo número F_p^n de vezes em que o predicado p aparece relacionada a n. Por fim, dividi-se esse valor pelo número de ocorrências F_p do predicado p em todo o grafo.

2.3. Métricas de Correlação

Como foram utilizadas muitas métricas de centralidade, existem várias possibilidades para calcular as relevâncias das relações. Precisa-se, de alguma maneira, decidir qual ou quais métricas utilizar. Descartar o resultado de uma métrica é descartar a semântica associada a técnica, e não é isso que buscamos.

Para isso, o Pytology permite correlacionar os valores gerados por mais de uma centralidade. Utilizado as métricas de Spearman[Zar 1998], Kendall[Abdi 2007] e Pearson[Sedgwick 2012] para correlacionamento de ranks. Spearman avalia relações monotônicas, lineares ou não; Pearson avalia relações lineares e Kendall que utiliza-se de avaliação ordinal.

3. Avaliação dos Experimentos

Os experimentos foram realizados executando o Pytology sobre um dataset de prêmios nobéis, disponível no Datahub.io, e executaram-se os cálculos de centralidade Betweenness(**B**), Closeness(**C**), Eigenvectors(**E**), Katz-Eigenvector(**K**) e Pagerank(**PR**). Como o Betwenness pode ser aplicado sobre arestas, foram feitos dois experimentos: o primeiro aplicando ele diretamente sobre as arestas e, em seguida, aplicando sobre os nós e calculando a distribuição. Chama-se essa segunda abordagem de Betwenness Distribuído(**BD**). Ordenaram-se as relações de acordo com suas relevâncias em forma de rank. A Tabela 1 apresenta as 5 relações mais relevantes de acordo com cada centralidade.

Tabela 1. Top 5 predicados mais relevantes de acordo com cada métrica

Posição	B	BD	C	E	K	PR
1	label	label	type	label	label	label
2	motivation	gender	year	motivation	year	year
3	gender	motivation	awardFile	gender	motivation	motivation
4	year	year	label	year	gender	gender
5	share	type	prizeFile	share	type	type

É evidente que as relações *label*, *type*, *year* e *motivation* estão bem colocadas nos ranks apresentados. Tais relacionamentos apresentam informações bem importantes sobre os dados: o rótulo do termo($label$), que pode ser um nome de país, pesquisador, prêmio, etc; o tipo do dado($type$), representando a classe que ele instancia; o ano($year$) e a motivação($motivation$) de um prêmio.

Dado o contexto dos prêmios nobéis, as informações *motivation*, que é a justificativa pelo recebimento do premio, e *year* são, de fato, bem relevantes. De um ponto de

vista mais voltado para o RDF, temos também que as informações *label* e *type* também são importantes, pois elas definem a representação textual de um elemento e o tipo de dado que ele é. As Tabelas 2, 3 e 4 referem-se às correlações entre os valores das centralidades obtidos de acordo com, respectivamente, Spearman, Pearson e Kendall.

Tabela 2. Correlação entre as relevâncias de acordo com Spearman

	C	E	B	BD	K	PR
C	1	0.729	0.751	0.770	0.823	0.793
E	0.729	1	0.890	0.946	0.843	0.847
B	0.751	0.890	1	0.914	0.806	0.790
BD	0.770	0.946	0.914	1	0.811	0.811
K	0.823	0.843	0.806	0.811	1	0.992
PR	0.793	0.847	0.790	0.811	0.992	1

Tabela 3. Correlação entre as relevâncias de acordo com Pearson

	C	E	B	BD	K	PR
C	1	0.294	0.257	0.354	0.334	0.307
E	0.294	1	0.971	0.981	0.989	0.957
B	0.257	0.971	1	0.991	0.985	0.990
BD	0.354	0.981	0.991	1	0.993	0.985
K	0.334	0.989	0.985	0.993	1	0.986
PR	0.307	0.957	0.990	0.985	0.986	1

Tabela 4. Correlação entre as relevâncias de acordo com Kendall

	C	E	B	BD	K	PR
C	1	0.575	0.586	0.630	0.646	0.624
E	0.575	1	0.739	0.838	0.733	0.740
B	0.586	0.739	1	0.769	0.673	0.642
BD	0.630	0.838	0.769	1	0.693	0.706
K	0.646	0.733	0.673	0.693	1	0.963
PR	0.624	0.740	0.642	0.706	0.963	1

Como Kendall basea-se em uma correlação ordinal, é esperado que os valores de relevância não sejam muito similares, uma vez que uma simples mudança na ordem do rank é suficiente para afetar a correlação. Quanto a Pearson e Spearman, ranks de técnicas semelhantes estão bem correlacionados, como Katz-Eigenvector, Pagerank e Eigenvectors e Betweeness e o Betweeness Distribuído. Esses resultados ajudam a fortalecer a ideia de que a distribuição das centralidades para as arestas mantém consigo a semântica da métrica utilizada. Percebe-se que há um certo direcionamento quando se trata de identificar os predicados mais importantes. Vale ressaltar que há casos de boas correlações com outras técnicas de semântica distinta, como evidência a tabela 3 na correlação de B com K.

4. Considerações Finais

Este trabalho apresenta uma proposta para calcular relevância de predicados em dados RDF a partir de métricas de centralidade. Aplicou-se essa técnica sobre uma grafo RDF de uma ontologia e obtiveram-se resultados demonstrando que utilizar as relevâncias de medidas de centralidade dos termos para calcular relevância das relações é algo que vale a pena ser explorado. Utilizar as métricas de correlação possibilita comparar os resultados dos ranqueamentos e permite analisar o direcionamento de relevância. Além de ser utilizado para validar o algoritmo de distribuição.

Pode-se também mencionar que não há técnica melhor ou pior, mas o que realmente é importante é saber o que se procura. Afinal cada métrica de centralidade possui uma semântica associada e ao utilizar as métricas de correlação pode-se entender quais predicados melhor atendem as técnicas correlacionadas. Permitindo então decidir qual predicado seria o mais relevante para um contexto especifico. Como trabalho futuro, pretende-se: utilizar a semântica de cada métrica e identificar o que seus valores representam no contexto de dados RDF, utilizar a semântica das relações para influenciar o calculo de relevância, usar o $schema$ do grafo RDF, se este o possuir, para efetuarmos um pré calculo sobre a relevância dos predicados.

Referências

Abdi, H. (2007). The kendall rank correlation coefficient. *Encyclopedia of Measurement and Statistics. Sage, Thousand Oaks, CA*, pages 508–510.

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. (2007). Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer.

Crubézy, M. and Musen, M. A. (2004). Ontologies in support of problem solving. In *Handbook on ontologies*, pages 321–341. Springer.

Elbassuoni, S. and Blanco, R. (2011). Keyword search over rdf graphs. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 237–242. ACM.

Freeman, L. C. (1978). Centrality in social networks conceptual clarification. *Social networks*, 1(3):215–239.

Marchionini, G. (2006). Exploratory search: from finding to understanding. *Communications of the ACM*, 49(4):41–46.

Mirizzi, R. and Di Noia, T. (2010). From exploratory search to web search and back. In *Proceedings of the 3rd workshop on Ph. D. students in information and knowledge management*, pages 39–46. ACM.

Musetti, A., Nuzzolese, A. G., Draicchio, F., Presutti, V., Blomqvist, E., Gangemi, A., and Ciancarini, P. (2012). Aemoo: Exploratory search based on knowledge patterns over the semantic web. *Semantic Web Challenge*, 136.

Roa-Valverde, A. J. and Sicilia, M.-A. (2014). A survey of approaches for ranking on the web of data. *Information Retrieval*, 17(4):295–325.

Sedgwick, P. (2012). Pearson's correlation coefficient. *BMJ: British Medical Journal (Online)*, 345.

Zar, J. H. (1998). Spearman rank correlation. *Encyclopedia of Biostatistics*.

Processamento de Consultas SPARQL em uma Base Relacional de Entidades

João G. Pauluk, Mariana M. Garcez Duarte, Rafael L. Prado e Carmem S. Hara

[1]Departamento de Informática – Universidade Federal do Paraná (UFPR)
{jgpauluk, marianamgd, rafaellimaprado426}@gmail.com, carmem@inf.ufpr.br

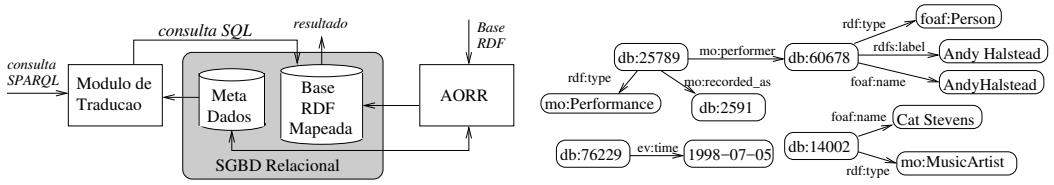
Abstract. *The huge volume of existing RDF datasets requires SPARQL queries to be efficiently processed. One approach to achieve this goal is to store RDF on a group-by-entity relational database, which explores structural similarity to group sets of triples in a single line of a relation. In this paper, we propose a method for translating SPARQL queries to SQL to be processed on such a database. Our experiments showed that the execution time of the translated queries are in average 250% lower, compared to queries on a triples relation.*

Resumo. *A grande quantidade de dados de RDF existente nos dias de hoje requer que as consultas SPARQL sejam processadas de forma eficiente. Uma possível abordagem para atingir tal objetivo é o armazenamento dos dados em uma base relacional de entidades, que explora a similaridade das estruturas para a horizontalização das triplas. Neste artigo, é proposto um método para a tradução de consultas SPARQL para SQL para ser processada sobre uma base relacional de entidades. Os experimentos realizados mostram que as consultas traduzidas e executadas sobre esta base obtiveram um ganho de desempenho de aproximadamente 250% em relação à consulta sobre uma tabela de triplas.*

1. Introdução

A *Web* semântica é uma extensão da *World Wide Web*, que promove a visão de dados interconectados e interpretáveis pelas máquinas. O W3C definiu o RDF como seu modelo de dados padrão e SPARQL como sua linguagem de consulta. Uma base RDF é composta por triplas SPO (sujeito, predicado e objeto), que pode ser representada na forma de um grafo, uma vez que o objeto de uma tripla pode ser o sujeito de outra. Uma consulta SPARQL é composta por padrões de triplas, que definem padrões de subgrafos a serem pesquisados na base. A grande quantidade de dados RDF existente requer que as consultas SPARQL sejam processadas de forma eficiente. Existem vários métodos para atingir tal objetivo [Aluç et al. 2014], sendo um deles o mapeamento dos dados RDF para o modelo relacional e a conversão das consultas SPARQL para SQL. Assim, é possível tirar proveito das otimizações que um SGBDR oferece ao utilizar a linguagem de consulta SQL. Tal método foi utilizado pelo Sistema de Armazenamento Otimizado de Dados RDF em SGBDR (AORR) [Prado et al. 2018].

O AORR explora a similaridade de estrutura dos dados para a horizontalização das triplas que compõem a base RDF. Esta estratégia de armazenamento é denominada neste trabalho de base relacional de entidades e visa melhorar o desempenho de consultas no formato estrela e flocos de neve [Aluç et al. 2014] através da diminuição do número de auto-junções necessárias para processá-las. Em [Prado et al. 2018] é proposto o algoritmo de extração de estrutura da base RDF, juntamente com os dados e metadados gerados. No



(a) Arquitetura do sistema (b) Grafo RDF

Figura 1. Arquitetura do sistema e Grafo RDF

entanto, o processo de tradução de consultas não é detalhado. Este artigo preenche esta lacuna, apresentando um algoritmo de tradução de consultas. A Figura 1a apresenta os componentes do sistema. O módulo de tradução de consultas é responsável por receber uma consulta SPARQL e traduzi-la para SQL baseado nos metadados de mapeamento, a fim de ser processada sobre a base relacional de entidades gerada pelo sistema AORR.

O restante do artigo está estruturado da seguinte forma. A Seção 2 apresenta trabalhos relacionados. O AORR é apresentado na Seção 3 e a Seção 4 detalha o algoritmo de tradução. A Seção 5 descreve a implementação, bem como uma análise experimental. A Seção 6 conclui o artigo enumerando alguns trabalhos futuros.

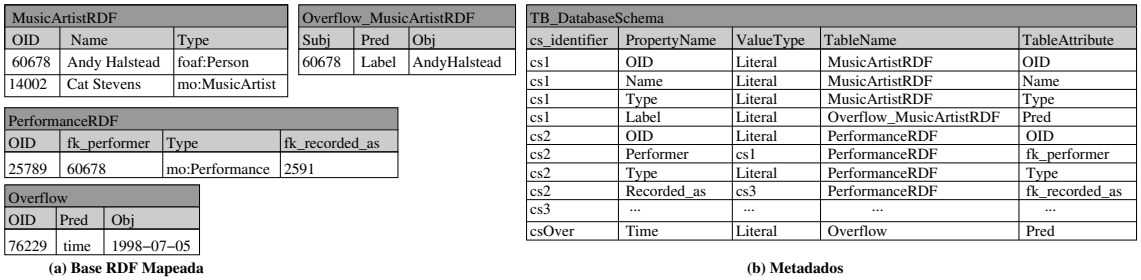
2. Trabalhos Relacionados

Existem diversas propostas para a tradução de consultas SPARQL para SQL. A proposta de [Chebotko et al. 2009], propõe um algoritmo genérico de tradução, na qual é criada uma subconsulta SQL para cada padrão de tripla da consulta SPARQL. Esta subconsulta é gerada a partir das funções α e β, que associam um padrão de tripla à relação na qual ela está armazenada no SGBD relacional e ao atributo, respectivamente. Estas subconsultas formam uma única consulta SQL com a utilização de operações como *inner join* para uma sequência de padrões ou *left outer join* para padrões opcionais. Outros trabalhos, tais como [Michel et al. 2016] e [Rodriguez-Muro et al. 2012] consideram que a base RDF foi gerada a partir de uma especificação R2RML [Das et al. 2012], que determina como uma base relacional foi mapeada para RDF. O processo de tradução utiliza esta informação para traduzir consultas SPARQL para SQL sobre a base relacional original. A proposta de tradução de consultas em [Chaloupka and Necasky 2016] é realizada a partir de um mapeamento entre os modelos definido pelo usuário.

O processo de tradução apresentado neste artigo é inspirado na proposta de [Chebotko et al. 2009]. Como [Chebotko et al. 2009], o artigo possui a ideia de subconsultas aninhadas de acordo com os tipos de junção entre cada padrão de tripla. No entanto, ao contrário das funções genéricas α e β, as funções que associam os padrões de triplas a tabelas e atributos são definidas a partir das tabelas de metadados gerados a partir do processo de extração de esquema do sistema AORR [Prado et al. 2018]. O AORR gera uma base relacional baseada em entidades, além de tabelas de overflow que armazenam triplas que não se adequam às estruturas das entidades.

3. Base Relacional de Entidades Gerada pelo AORR

O Sistema AORR [Prado et al. 2018] possui um módulo de extração de estrutura de uma base RDF, que tem por objetivo identificar tipos de entidades, ou seja, conjuntos de sujeitos que possuem estruturas similares. Para cada entidade é criada uma tabela com seus



MusicArtistRDF

OID	Name	Type
60678	Andy Halstead	foaf:Person
14002	Cat Stevens	mo:MusicArtist

Overflow_MusicArtistRDF

Subj	Pred	Obj
60678	Label	AndyHalstead

PerformanceRDF

OID	fk_performer	Type	fk_recorded_as
25789	60678	mo:Performance	2591

Overflow

OID	Pred	Obj
76229	time	1998–07–05

(a) Base RDF Mapeada

TB_DatabaseSchema

cs_identifier	PropertyName	ValueType	TableName	TableAttribute
cs1	OID	Literal	MusicArtistRDF	OID
cs1	Name	Literal	MusicArtistRDF	Name
cs1	Type	Literal	MusicArtistRDF	Type
cs1	Label	Literal	Overflow_MusicArtistRDF	Pred
cs2	OID	Literal	PerformanceRDF	OID
cs2	Performer	cs1	PerformanceRDF	fk_performer
cs2	Type	Literal	PerformanceRDF	Type
cs2	Recorded_as	cs3	PerformanceRDF	fk_recorded_as
cs3	...	...	...	...
csOver	Time	Literal	Overflow	Pred

(b) Metadados

Figura 2. Base relacional gerada pelo AORR

predicados. Dada a natureza não estruturada das bases RDF, algumas triplas não se adequam ao esquema relacional extraído. Assim, o AORR cria um conjunto de tabelas de triplas (SPO) para armazenar tais informações. Estas tabelas são chamadas de *overflow*. As tabelas de overflow de entidades armazenam predicados infrequentes de sujeitos armazenados na tabela de entidade correspondente, como por exemplo, `MusicArtistRDF` e `Overflow_MusicArtistRDF`. Além disso, há sujeitos que não pertencem a nenhum tipo de entidade. Eles são armazenados na tabela *Overflow*. A Figura 2a apresenta a base relacional gerada pelo AORR a partir da base RDF ilustrada na Figura 1b.

Para permitir a tradução de consultas SPARQL para SQL sobre a base gerada, o AORR armazena informações sobre o mapeamento em tabelas de metadados. As principais são : `TB_Subj_OID`, `TB_FullPredicate` e `TB_DatabaseSchema`. A tabela `TB_Subj_OID` associa IRIs a identificadores (OID) numéricos. A tabela `TB_FullPredicate` associa IRIs de predicados ao nome de uma coluna em uma tabela de entidade ou a um label utilizado como valor da coluna predicado nas tabelas de overflow. Por exemplo, a IRI `http://purl.org/ontology/mo/recorded_as` é associada ao label `fk_recorded_as` da tabela `PerfomanceRDF`. Já a tabela `TB_DatabaseSchema` traz informações sobre o mapeamento, como mostra a Figura 2b. Cada tipo de entidade é identificado por um `cs_identifier`, que possui um predicado `PropertyName` com um valor do tipo `ValueType`, que pode ser um literal ou uma IRI que pertence a uma outra entidade. O predicado é armazenado na tabela `TableName` na coluna `TableAttribute`. Observe que uma entidade pode ter predicados armazenados na tabela de entidade como no seu overflow (como é o caso de `cs1` no Exemplo da Figura 2b). Neste artigo considera-se que cada predicado encontra-se ou na tabela de entidade ou no seu overflow, mas não em ambos. Além disso, todos os predicados em sujeitos na tabela `Overflow` possuem o mesmo valor `csOver` para o atributo `cs_identifier` da tabela `TB_DatabaseSchema`.

4. Tradução de Consultas SPARQL para SQL

Este artigo considera consultas formadas por padrões de grafos básicos (BGP - *Basic Graph Patterns*), nas quais os padrões de triplas são da forma *(variável, IRI, variável)*. O Algoritmo 1 apresenta como é realizada a tradução de uma consulta SPARQL Q para SQL. Primeiro, cada variável no resultado é associada a um rótulo (Linha 2) e é realizada uma busca dos identificadores de cada IRI de predicado na tabela *TB_FullPredicate* (Linha 3). Os demais passos são detalhados a seguir.

Procura por padrões estrela (Linhas 5-8). Neste passo os padrões de tripla em Q que possuem o mesmo sujeito são agrupados, ou seja, são identificados os padrões estrela na

Algoritmo 1: Algoritmo de Tradução

Entrada: Consulta SPARQL Q = select $?x_1...?x_n$ where PT
Saída: Consulta SQL Q'

```
1  início
2      Associa cada variável x_1, ..., x_n a rótulos name(x_1), ..., name(x_n);
3      Busca em TB_FullPredicate os IDs das IRIs de predicados em PT;
4      Seja V = {v_1, ...v_p} o conjunto de sujeitos (variáveis) em PT;
5      para cada variável v ∈ V faça
6          entidades(v) := busca em TB_DatabaseSchema entidades cs que
7                          possuem todos os predicados de v em PT;
8      fim
9      para cada variável v ∈ V faça
10         filtra( entidades(v) );
11     fim
12     condJuncao := extraiCondJuncao(entidades(v_1), ..., entidades(v_p));
13     para cada variável v faça
14         para cada entidade cs em entidade(v) faça
15             sql(v, cs) := geraSubConsulta(v, cs);
16         fim
17         sql(v) := sql(v, cs_1) UNION ...UNION sql(v, cs_j);
18     fim
19     Q':= SELECT name(x_1), ... name(x_n)
20           FROM sql(v_1), ...sql(v_p) WHERE condJuncao;
21 fim
```

consulta. A partir deles, é realizada uma busca na tabela `TB_DatabaseSchema` para determinar quais entidades possuem todos os predicados existentes em cada padrão estrela. Considere por exemplo, a consulta SPARQL na Figura 3a. O sujeito $?a$ é associado à entidade com identificador $cs1$ e o sujeito $?b$ à entidade $cs2$. Além de restringir o processo de tradução às entidades $cs1$ e $cs2$, a tabela `TB_DatabaseSchema`, possui ainda as informações da tabela, atributo e tipo associado a cada predicado.

Filtragem por ligações (Linhas 9-12). Os tipos dos predicados são utilizados para filtrar ligações sujeito-objeto entre diferentes padrões estrela. Para ilustrar, considere o padrão de tripla *?b performer ?a* da Figura 3a . Como $?a$ é sujeito de outras triplas, há uma ligação sujeito-objeto. Assim, $cs1$, entidade relacionada à $?a$ deve corresponder ao tipo do predicado *fk_performer* de $cs2$. Com a verificação dos tipos dos objetos, é possível filtrar as entidades associadas a cada sujeito. O mesmo tipo de filtragem é utilizado para ligações objeto-objeto. Estas ligações dão origem às condições de junção entre padrões estrela (Linha 24 da Figura 3b).

Montagem do comando SQL (Linhas 13-20). Para cada variável sujeito é gerada uma subconsulta SQL. Essas subconsultas são inseridas na cláusula FROM da consulta final (Linha 20). Quando houver mais de uma entidade associada a uma mesma variável, as subconsultas geradas para cada variável são colocada em uma única expressão com a operação de UNION (Linha 17). A geração da subconsulta para cada entidade relacio-

```
SELECT ?n ?t ?l ?b
WHERE {
      ?a name ?n .
      ?a type ?t .
      ?a label ?l .
      ?b performer ?a .
}
```

(a) Consulta SPARQL

```
1 SELECT a.name248558 AS n,
2          a.type8HG5ET AS t,
3          a.labelYGC7VX AS l,
4          b.Subj AS b
5 FROM
6 (SELECT a1.OID,
7          a1.Name AS name248558,
8          a1.Type AS type8HG5ET,
9          o1.Obj AS labelYGC7VX
10 FROM MusicArtistRDF a1,
11      Overflow_MusicArtistRDF o1
12 WHERE a1.Name IS NOT NULL
13       AND a1.Type IS NOT NULLL
14       AND o1.Pred = 'label'
15       AND o1.Obj IS NOT NULL
16       AND a1.OID = o1.OID) a,
17 (SELECT b1.OID,
18          s.Subj,
19          b1.fk_performer AS fk_performer0SZFR6,
20 FROM PerformanceRDF b1,
21      TB_Subj_OID s,
22 WHERE b1.fk_performer IS NOT NULL
23       AND b1.OID = s.OID) b
24 WHERE a.OID = b.fk_performer0SZFR6
```

(b) Consulta SQL

Figura 3. Consulta SPARQL traduzida em consulta SQL

nada a uma variável tem as seguintes linhas gerais: **(a)** para cada entidade: a consulta utiliza a tabela de entidade correspondente e possivelmente múltiplas vezes sua tabela de overflow específico, que são combinadas pela operação de junção sobre o atributo OID. Por exemplo, para a variável $?a$, é gerada a junção das tabelas `MusicArtistRDF` e `Overflow_MusicArtistRDF` (Linhas 6-16 da Figura 3b); **(b)** para o Overflow: são realizadas múltiplas auto-junções da tabela `Overflow` sobre o atributo OID, uma vez para cada predicado da entidade *CSover* na tabela `TB_DatabaseSchema`; **(c)** caso a própria variável sujeito estiver no resultado é necessário fazer uma junção com a tabela `TB_Subj_OID` para obter a IRI relacionada ao OID (Linhas 17-23 da Figura 3b).

5. Análise Experimental

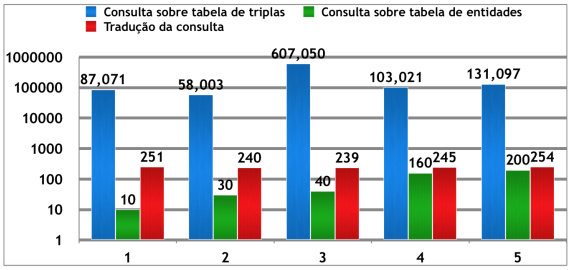
O algoritmo de tradução foi implementado utilizando a linguagem Python, a ferramenta de *parsing* ANTLR[1] e o SGBD MySQL. Nesta seção são relatados experimentos realizados para determinar o tempo de tradução das consultas e comparar o tempo de execução das consultas traduzidas com o tempo de execução sobre uma tabela de triplas SPO. O computador utilizado para executar os experimentos foi um MacOSX Intel Core m3 1.1 GHz com 8 GB de memória RAM. O banco de dados utilizado possui 26 tabelas, que foram geradas com o sistema AORR, a partir da base RDF Peel[2].

Os testes executados consistiram de 5 consultas. Os testes realizados foram baseados em três padrões estrelas presentes na base Peel, como detalhado na Figura 4b. As consultas contém os seguintes padrões: Consulta 1: PE1; Consulta 2: PE2; Consulta 3: PE3; Consulta 4: PE1 e PE2; Consulta 5: PE1, PE2 e PE3[3]. A Base SPO tem tamanho de 44.58 MB, enquanto a base estruturada pelo AORR tem 21.84 MB. Foram criados índices na tabela SPO sobre o predicado e sujeito, enquanto as tabelas de entidades tem índices apenas sobre o atributo OID. O impacto da criação dos índices foi apresentada anteriormente em [Duarte and Hara 2018], que também explorou uma forma alternativa de tradução de consultas baseada em visões. O tempo de processamento das consultas, bem como o tempo de tradução, reportados em milisegundos, estão ilustrados na Figura 4a. Devido a diferença nos valores, foi utilizada uma escala logarítmica. É possível perceber que o tempo de tradução das consultas permanece praticamente constante e que seu custo é alto, comparado ao tempo de processamento da consulta. Isso se deve às diversas junções sobre a tabela `TB_DatabaseSchema` executadas pelo algoritmo de

[1] `http://www.antlr.org/`

[2] `http://dbtune.org/bbc/peel/`

[3] As consultas estão disponíveis em `http://www.inf.ufpr.br/mmgduarte/SBBD18/`



(a) Tempo de Execução das consultas

Padão Estrela	Padrões Utilizados
PE 1	?a ev:place ?p ?a mo:produced_signal ?s ?a rdf:type ?t
PE 2	?s mo:published_as ?x ?s rdf:type ?y ?s rdfs:label ?l
PE 3	?x elem:title ?b ?x rdf:type ?c ?x rdfs:label ?d

(b) Caracterização das consultas

Figura 4. Resultado dos Experimentos

tradução. No futuro, é planejado explorar a utilização de índices sobre esta tabela e uma representação em memória. O tempo de execução das consultas sobre as tabelas de entidades apresenta um ganho significativo em relação às consultas sobre a base no formato SPO. Isso se deve ao volume da tabela de triplas, que utiliza IRIs completas, além da quantidade de auto-junções no processamento de consultas. Contabilizando o tempo da tradução e processamento, as tabelas de entidades apresentam uma redução no tempo de processamento de aproximadamente 250%. A consulta C3 apresenta um comportamento diferenciado, com um ganho de 2175% devido a quantidade de dados envolvidos.

6. Conclusão

Este artigo apresentou um algoritmo de tradução de consultas SPARQL para SQL sobre uma base relacional de entidades geradas com o sistema AORR. O algoritmo foi implementado e os resultado experimentais mostraram que o tempo de processamento das consultas SPARQL teve uma redução de aproximadamente 250% quando comparado a uma base relacional de triplas. Como trabalho futuro é planejado estender o algoritmo para consultas SPARQL mais expressivas e a otimização do processo de tradução.

Referências

Aluç, G., Ozsu, M. T., and Daudjee, K. (2014). Workload matters: Why rdf databases need a new design. *Proc. of the Int. Conf. on Very Large Data Bases*, 7(10):837–840.

Chaloupka, M. and Necasky, M. (2016). Efficient sparql to sql translation with user defined mapping. In *Proc. of the Knowledge Engineering and Semantic Web Conference*.

Chebotko, A., Lu, S., and Fotouhi, F. (2009). Semantics preserving sparql-to-sql translation. In *Data Knowledge Engineering*, pages 973–1000. Volume 68 Issue 10.

Das, S., Sundara, S., and Cyganiak, R. (2012). R2rml: Rdb to rdf mapping. `http://www.w3.org/TR/r2rml/`.

Duarte, M. M. G. and Hara, C. S. (2018). Otimização do mapeamento de consultas SPARQL para SQL. In *Escola Regional de Banco de Dados*.

Michel, F., Zucker, C. F., and Montagnat, J. (2016). A generic mapping-based query translation from sparql to various target database query languages. In *Proc. of the 12th International Conference on Web Information Systems and Technologies*.

Prado, R. L., Schroeder, R., and Hara, C. S. (2018). Armazenamento otimizado de dados RDF em um SGBD relacional. In *Proc. of the Brazilian Symposium on Databases*.

Rodriguez-Muro, M., Hardu, J., and Calvanese, D. (2012). Quest: Effcient sparql-to-sql for rdf and owl. In *Proc. of the ISWC 2012 Posters Demonstrations Track (ISWC-PD)*.

GovDadosMB: Um framework de Governança de Dados Corporativos para a Marinha do Brasil

Marta Rigaud Faria[1], Madalena Lopes e Silva[1], Kelli de Faria Cordeiro[1,2]

[1]Seção de Engenharia da Computação – Instituto Militar de Engenharia (IME)
Praça General Tibúrcio 80, Praia Vermelha – 22.290-270 – Rio de Janeiro – RJ – Brasil

[2]Centro de Análise de Sistemas Navais
Ed. 23 do AMRJ - R. da Ponte, s/n, Centro – 20.091-000 – Rio de Janeiro - RJ – Brasil

martarigaud@yahoo.com.br, madalena@marinha.mil.br, kelli@marinha.mil.br

Abstract. *The increasing importance of data in supporting the decision-making process makes it essential to orchestrate data management activities, especially in corporate environments. Data Governance frameworks present guidelines that structure the execution of such activities, but require adaptation to the particular complexity of each corporate institution. This paper presents a comparative analysis between data governance frameworks. Based on the result of the analysis, this article proposes a framework adapted to the Navy, called GovDadosMB, with a focus on the interoperability of personnel management data.*

Resumo. *A crescente importância dos dados no apoio ao processo de tomada de decisão torna imprescindível a orquestração das atividades de gestão de dados, especialmente, em ambientes corporativos. Os frameworks de Governança de Dados apresentam diretrizes que estruturam a execução de tais atividades, contudo requerem uma adaptação à complexidade particular de cada instituição corporativa. Este paper apresenta uma análise comparativa entre frameworks de governança de dados. Com base no resultado da análise, este paper propõe um framework adaptado à Marinha do Brasil, chamado GovDadosMB, com foco na interoperabilidade dos dados de gestão de pessoal.*

1. Introdução

A Marinha do Brasil (MB) atua no aprestamento das Forças Navais e no emprego dessas forças na defesa do território nacional no mar e nas águas interiores. Para garantir o cumprimento de sua missão, a MB tem uma estrutura organizacional hierarquizada composta por centenas de organizações militares distribuídas por todo país que utilizam centenas de sistemas de informação para apoiar a operacionalização de suas atividades.

Com a evolução do uso dos dados, o potencial da organização para alcançar as suas metas e encontrar novos caminhos para inovar baseia-se principalmente em dados. As instituições estão cada vez mais conscientes de que quanto maior nível de qualidade dos dados maiores serão os benefícios que poderão obter [Carretero et al. 2017].

Diante dos desafios do uso dos dados, a Governança de Dados Corporativos (GD) torna-se essencial na definição de políticas e procedimentos para assegurar o gerenciamento proativo e efetivo de dados. Complementarmente, a adoção de um framework de GD permite a colaboração entre diversos níveis organizacionais para o gerenciamento de seus dados, além de apoiar o alinhamento da gestão de dados com seus objetivos corporativos [Cheong e Chang 2007].

A partir de uma visão corporativa, esse trabalho apresenta os frameworks de GD utilizados como base da proposta do GovDadosMB. Para tal, a seção 2 descreve e compara os frameworks analisados e a seção 3 descreve o GovDadosMB.

2. Frameworks de Governança de Dados

Os frameworks ajudam a explicitar ideias e facilitam a comunicação de conceitos complicados ou ambíguos, possibilitando torná-los claros e objetivos [DGI 2014]. Segundo Khatri e Brown (2010), para implantar uma GD é necessário identificar quais são as decisões fundamentais e quem deveria tomá-las no processo de tomada de decisão. De forma complementar, o Data Governance Institute (DGI) sugere realizar uma GD usando as perspectivas *Who - What - When - Where – Why*. Seguindo esta linha, a International Business Machines (IBM) propôs um modelo, baseado no Capability Maturity Model (CMM), composto por dez etapas necessárias e quatro etapas opcionais. Além desses, o framework da Data Management Association (DAMA) propõe a estruturação da GD em dez áreas de conhecimento compiladas no DMBOK, sendo este um dos frameworks mais completos atualmente [DAMA 2017].

Como pode ser observado, vários frameworks foram desenvolvidos para atender necessidades variadas. Para tanto, possuem um série de características associadas, conforme descrito na Tabela 1. As características presentes e ausentes em cada um dos frameworks foram utilizadas para compará-los com o objetivo de subsidiar a concepção de um framework para MB.

Tabela 1. Comparação entre frameworks de GD

Característica	Papéis e Responsa-bilidades	Políticas e Padrões	Estrutura Formal de GD	Avaliação da Maturidade	Qualidade dos dados	Monitora-mento de conformi-dade (ciclo)	Gestão do conheci-mento	Segurança e Privacidade
Khatri e Brown	X	X			X		X	X
DGI	X	X	X	X	X	X		
IBM	X		X	X	X	X	X	X
DAMA	X	X	X	X	X	X		X

Pode-se observar que os frameworks listados possuem algumas das características desejáveis, mas nenhum deles possui todas as características. O modelo proposto por Khatri e Brown (2010) foi criado com o principal objetivo de estabelecer quem detém os direitos de decisão e quem é o responsável pela tomada de decisões sobre os ativos de dados de uma organização, sem se preocupar em estabelecer formalmente uma estrutura de GD. Tal fato se deve à preocupação com o

estabelecimento de uma ligação entre o negócio e os dados através da definição de políticas e padrões, em comunicar aos *stakeholders* qual a estratégia de governança está sendo adotada, bem como definir os requisitos para utilização dos dados.

A proposta do DGI foi apresentar um framework prático que pudesse apoiar a comunicação dos *stakeholders* com maior clareza e direcioná-los para a definição de um programa de GD [DGI 2014]. O modelo contempla as características desejáveis de um framework, exceto a gestão de conhecimento.

O Processo Unificado de Governança da IBM fornece um conjunto de marcos para ajudar as organizações de todos os tamanhos a medir a forma como regem seus dados [Were e Moturi 2017]. As partes integrantes deste processo são: a avaliação de maturidade, a definição de papéis e responsabilidades, o estabelecimento de métricas, o ciclo de vida da informação e o gerenciamento da qualidade.

No framework proposto pelo DAMA, o estabelecimento de padrões e políticas, papéis e responsabilidade é crucial. Dentro deste contexto o DAMA sugere a adoção de uma estrutura formal de GD para exercer o controle da gestão de dados. Além disso, há áreas de conhecimento específicas para tratar da qualidade dos dados, da gestão de armazenamento e operação de dados e da gestão de segurança de dados. Por fim, em cada área de conhecimento são definidas métricas que permitem a avaliação da governança de forma abrangente.

A adoção de frameworks de GD no governo federal está presente em diversas instituições como o Banco Central do Brasil que estrutura a sua Governança de Informações em objetivos, princípios, diretrizes e responsabilidades [BCB, 2013]. O Tribunal de Contas da União também destaca a importância da GD nos órgãos de controle [Stumpf 2016].

Da mesma forma, a MB prioriza a definição de papéis e responsabilidades em uma estrutura organizacional formal de GD para elevar o nível de maturidade dos processos de gestão de dados por área de conhecimento visando o apoio a tomada de decisão. Baseada nessas prioridades, a instituição escolheu reunir as características dos frameworks descritos que se encaixam aos requisitos demandados para propor o GovDadosMB.

3. GovDadosMB: framework de Governança de Dados Corporativos da Marinha do Brasil

Devido à amplitude e complexidade da implantação da GD, uma abordagem cíclica e evolutiva foi adotada, iniciando com a área de gestão de pessoal por ser uma das áreas da governança corporativa com maior densidade de informações.

A gestão de pessoal na instituição envolve administrar dados biográficos, de carreira e financeiros de milhares de servidores civis e militares ativos, veteranos, seus dependentes e beneficiários de pensão. Para esse fim, as organizações empregam diferentes processos que são automatizados por centenas de sistemas de informação com as suas respectivas bases de dados. Tais sistemas interoperam entre si por meio de cargas, serviços de dados, acessos direto às bases de dados, assim como ainda carga de dados via job submetido em ambiente de mainframe, conforme visão geral ilustrada na Figura 1.

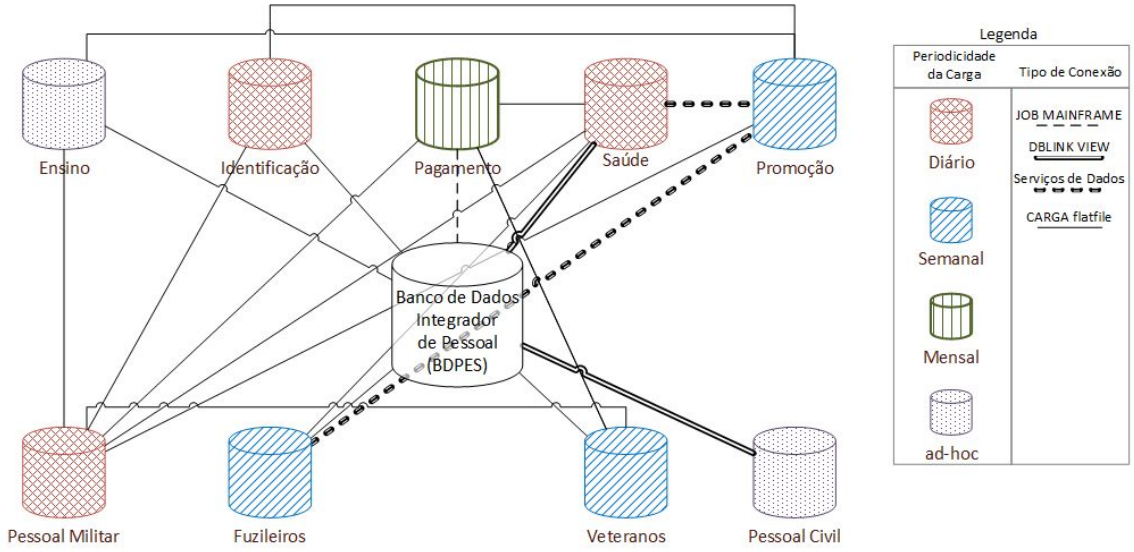


Figura 1. Interoperabilidade entre os sistemas envolvidos na Gestão de Pessoal da MB

A ampliação da automatização dos processos de negócio em um cenário corporativo, como na MB, ocasiona o crescimento desordenado dos dados gerando problemas como redundância não controlada, falta de padronização, dados inconsistentes, dentre outros. Além disso, há uma dificuldade em administrar os dados provenientes de processos interorganizacionais de modo que possam apoiar o consumo de dados dos decisores que precisam de uma visão de estratégica sobre o emprego e a mobilização da força de trabalho militar bem como o planejamento atuarial.

Diante deste cenário, a MB percebeu a necessidade de implementar a GD, para ampliar a visão de Governança de Tecnologia da Informação e Comunicação (TIC), estruturando seus princípios nos níveis estratégico, tático e operacional. Com isso, é possível apoiar a Governança Corporativa com uma visão integrada e confiável dos dados resultando em obtenção de informações necessárias à tomada de decisão.

A partir dos frameworks apresentados e do cenário da gestão de dados da MB, o framework GovDadosMB foi proposto, conforme ilustrado na Figura 2, com uma abordagem cíclica e evolutiva. A exemplo dos framework DGI, IBM e DAMA, foram considerados aspectos sociológicos e culturais intrínsecos às corporações de grande porte. O GovDadosMB é baseado em princípios norteadores que, segundo [Katri e Brown 2007], estabelece a direção de todas as outras decisões e, no caso da MB, direcionam ações das diversas áreas de conhecimento propostas pelo DAMA.

Dentre as áreas de conhecimento sugeridas pelo DAMA, na implementação do GovDadosMB, quatro foram priorizadas para ampliar as ações executivas visando elevar seu nível de maturidade: Integração e Interoperabilidade, Desenvolvimento de Dados, Dados Mestres e de Referência, e Metadados. Tal priorização foi motivada pelo cenário identificado na gestão dos dados de pessoal.

O primeiro passo para a implantação um programa formal de GD foi criação de uma estrutura organizacional que pudesse exercer a autoridade e controle sobre os ativos de dados. Partindo dessa premissa, um Comitê Técnico de Governança de Dados

Corporativos (COTEC-Dados) composto por Gestores de Dados Locais e Curadores do Negócio foi estruturado.

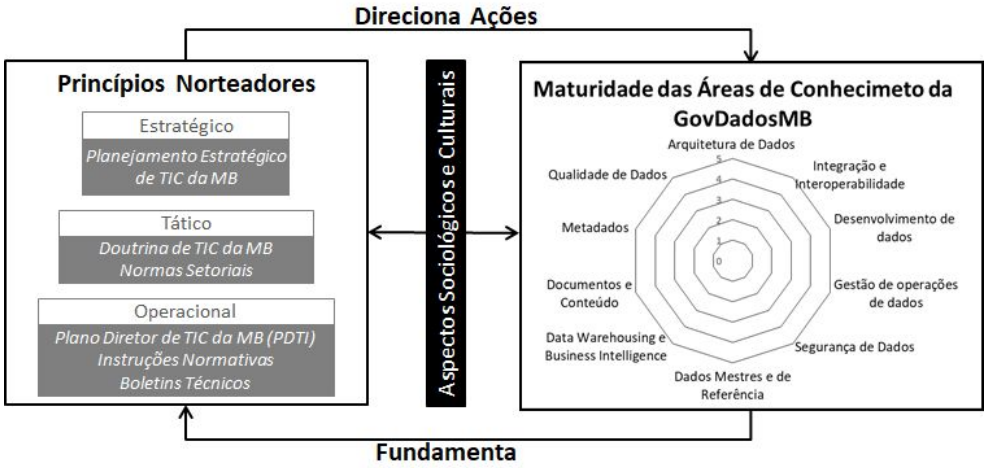


Figura 2. Framework GovDadosMB

O COTEC-Dados tem como objetivo minimizar os impactos sociológicos e culturais naturais de uma gestão de dados corporativos realizados de forma descentralizada por muitos anos. Para isso, esforços têm sido envidados para que o COTEC-Dados seja presidido por um órgão hierarquicamente superior a todos os outros e que faça parte da Doutrina de TIC da MB que é um princípio norteador a nível tático.

O segundo passo para a implantação da GovDadosMB é a definição do Modelo de Dados Corporativo composto por Dados Mestre e de Referência que, no momento, foca em três entidades: Pessoal, Organização Militar e Meio Operativo. Na sequência, os metadados dessas entidades serão centralizados e compartilhados pelos respectivos gestores de dados, criando-se, assim um repositório de metadados. Atualmente, por prática, os gestores descrevem seus dados nos atributos *description* e *comments*, recurso disponível nos principais Sistemas Gerenciadores de Banco de Dados. Tal prática irá facilitar a alimentação do repositório de metadados corporativo.

As ações executivas descritas, envolvendo diversas áreas de conhecimento, irão permitir o aumento do nível de maturidade dos seus respectivos processos e de outros, como os processos da área de Integração e Interoperabilidade minimizando as redundâncias e as inconsistências. Como consequência, será possível apoiar a tomada de decisão utilizando dados com maior nível de qualidade.

Por fim, a partir da análise do nível de maturidade dessas áreas, será possível fundamentar novos princípios norteadores que irão direcionar as ações executivas do ciclo seguinte evidenciando a efetividade da GovDadosMB.

4. Conclusão

O framework GovDadosMB proposto neste artigo foi concebido com base na análise comparativa de diversos frameworks, com os quais foram feitas adaptações para orquestrar as atividades de gestão de dados com foco na interoperabilidade dos sistemas de gestão de pessoal da MB visando o apoio a tomada de decisão.

O GovDadosMB se destaca pelo alinhamento entre os princípios norteadores da

GovTIC e com as ações de GD, levando em consideração os aspectos sociológicos e culturais da MB. A implantação da GovDadosMB, mesmo em fase inicial, apresentou resultados positivos ao se estabelecer uma estrutura organizacional formal, a qual definiu os papéis e as responsabilidades aos *stakeholders.*

Como trabalho futuro, será realizado um aprofundamento sistemático nos frameworks de GD e de trabalhos da literatura para que a GovDadosMB possa ser aprimorada com o detalhamento das áreas de conhecimento, em especial, a interoperabilidade e a integração entre sistemas de informação.

5. Referências

BRASIL, BANCO CENTRAL DO BRASIL – BCB. Portaria nº 47, de 20 de fevereiro de 2013. Diário Oficial da União – DOU, 21 de fevereiro de 2013. Seção 1, p. 24.

Carretero, A. G., Gualo , F., Caballero, I. and Piattini, M. (2017), "MAMD 2.0: Environment for data quality processes implantation based on ISO 8000-6X and ISO/IEC 33000", Computer Standards & Interfaces 54 (2017) 139–151

Cheong, Lai Kuan and Chang, Vanessa, "The Need for Data Governance: A Case Study" (2007). ACIS 2007 Proceedings. 100. http://aisel.aisnet.org/acis2007/100

DAMA International (2017), "DAMA-DMBOK: Data Management Body of Knowledge", 2nd Edition.

Friedman, T (2006), "Key Issues for Data Management and Integration", Gartner Research. ID Number: G00138812, March 2006.

Khatri, V. and Brown, C.V. (2010), "Designing data governance", Communications of the ACM, Vol. 53 No. 1, pp. 148-152.

Stumpf, R. D., "O porquê de governança de dados em organizações de controle", Revista do Tribunal de Contas da União, No. 137(2016), pp. 107-116, disponível em: http://revista.tcu.gov.br/ojs/index.php/RTCU/issue/view/68.

Sunir Soares, MC Press, LLC, 2010, "The IBM Data Governance Unified Process: Driving Business Value with IBM Software and Best Practices".

The Data Governance Institute (2014), "How to use the DGI Data Governance framework to configure your program", The Data Governance Institute, available at: http://www.datagovernance.com/wp-content/uploads/2014/11/wp_how_to_use_the_dgi_data_governance_framework.pdf, Acessado em 22 Mai 2018.

Were, V. and Moturi, C. (2017), "Toward a data governance model for the Kenya health professional regulatory authorities", The TQM Journal Vol. 29 No. 4, 2017 pp. 579-589.

LinkedECG: Uma Abordagem para a Integração e Publicação de Dados de Eletrocardiograma

Douglas Torquato[1], Daniel Rodrigues[1], José Maria Monteiro[1], João Paulo Madeiro[2], Angelo Brayner[1], Vânia Vidal[1], Narciso Arruda[1], Tiago Vinuto[1]

[1]MDCC – Universidade Federal do Ceará (UFC)
Fortaleza – CE – Brasil

[2]IEDS – UNILAB
Redenção, CE – Brasil

{douglas,daniel,monteiro,brayner,vvidal,narciso,tiagosv}@lia.ufc.br

jpaulo.vale@unilab.edu.br

Abstract. *The electrocardiogram (ECG) is a widespread medical procedure in the cardiology field due to the fact that it is a fast, low-cost and non-invasive examination, and its analysis allows anomalies to be detected and interpreted by health experts. However, the data that comprise or may be extracted from the ECG signals are quite complex, heterogeneous and do not follow a single standardization. In this context, the present work proposes an approach, named LinkedECG, for ECG feature extraction (resulting from signal processing), ECG data integration and publishing following the main standards related to open data sharing on the web. The proposed methodology yields that data extracted from a set of collected signals and/or from publicly available signal records be integrated and published, in order to support, using Web environment, complex queries, execution of mining algorithms, and also enable collaboration among specialists.*

Resumo. *O eletrocardiograma (ECG) é um exame bastante difundido na área de cardiologia, devido ao fato de ser um procedimento simples, de baixo custo e não-invasivo. O seu estudo permite que doenças e anomalias cardíacas possam ser detectadas por especialistas da área. Contudo, os dados que compõem ou que podem ser extraídos dos sinais ECG são bastante complexos, heterogêneos e não seguem uma única padronização. Neste contexto, este trabalho propõe uma abordagem, denominada LinkedECG, para a extração de parâmetros decorrentes do processamento do sinal, integração e publicação de dados de ECG, seguindo os principais padrões para o compartilhamento de dados abertos na Web. A metodologia proposta permite que dados extraídos de um conjunto de sinais coletados e/ou disponíveis em bases de dados públicas sejam integrados e publicados, de forma a oferecer suporte, no próprio ambiente Web, a consultas complexas, execução de algoritmos de mineração, além de possibilitar a colaboração entre especialistas.*

1. Introdução

A Eletrocardiografia é uma técnica utilizada para registrar as alterações de potencial elétrico produzidas pela atividade cardíaca. A evolução temporal das referidas alterações é chamada de sinal eletrocardiograma (ECG) [Geselowitz 1989], o qual é o teste mais difundido na Cardiologia para o diagnóstico de doenças e anomalias cardíacas [Gonçalves et al. 2007]. A análise do comportamento do sinal ECG permite extrair informações diversificadas, as quais podem subsidiar a identificação de uma grande variedade de doenças cardíacas.

Devido ao grande interesse no estudo e interpretação do conteúdo do sinal ECG, diversos padrões de armazenamento dos dados foram criados [Gonçalves et al. 2007]. Esses padrões têm por objetivo: (1) permitir o armazenamento de arquivos de sinais ECG em prontuários eletrônicos (EHR); (2) facilitar a comunicação/intercâmbio dos resultados de exames cardíacos entre diferentes profissionais e/ou hospitais [Gonçalves et al. 2007]. Dentre os principais padrões criados, destacam-se: (i) AHA/MIT-BIH (Physionet) [Goldberger et al. 2000]; (ii) SCP-ECG [Mandellos et al. 2010] e (iii) HL7 aECG [Bond et al. 2011]. Apesar desses padrões tratarem de um mesmo domínio do conhecimento, os conceitos adotados são heterogêneos e distintos entre si. Este cenário dificulta, portanto, a integração de dados de ECG que estão armazenados em padrões diferentes.

Por outro lado, o estudo sobre a Web de Dados vem crescendo a cada ano. Essa nova Web se baseia em um conjunto de melhores práticas para publicação e consumo de dados estruturados, mais conhecido como *Linked Data*, possibilitando a integração entre itens de diferentes bases de dados para formar um único espaço de dados global [Heath and Bizer 2011]. As premissas de *Linked Data* fundamentam-se nas tecnologias da Web Semântica e permitem reduzir a complexidade de integrações devido às ligações estabelecidas e descritas entre os conjuntos de dados. O uso de um modelo de dados padronizado (RDF) e um mecanismo padronizado de consulta (linguagem de consulta SPARQL) simplificam ainda mais a integração de dados.

O principal objetivo deste trabalho consiste em propor uma abordagem de extração automática de parâmetros, decorrentes do processamento de sinais ECG, integração e publicação de dados de sinais ECG no formato RDF, com base nas tecnologias da Web Semântica e nos padrões de *Linked Data*. Esta abordagem possibilita a realização de consultas na linguagem SPARQL, bem como a execução de inferências sobre os dados.

O artigo está estruturado como segue. Na seção 2, são discutidos os trabalhos relacionados. Na seção 3, são detalhadas as diferentes etapas da abordagem proposta. Na seção 4, é apresentado um estudo de caso com a finalidade de avaliar a abordagem proposta. Finalmente, na seção 5, são apresentadas as conclusões e trabalhos futuros.

2. Trabalhos Relacionados

Gonçalves [Gonçalves et al. 2007] propôs uma ontologia para representar sinais de ECG, a qual independe da aplicação e da linguagem de codificação. A ontologia proposta é descrita através da linguagem OWL (Web Ontology Language). Adicionalmente, o autor descreve como mapear cada classe de cada um de diferentes formatos de dados ECG (AHA/MIT-BIH, SCP-ECG e FDA XML/HL7 aECG) com a correspondente classe na ontologia proposta. Raimond L. Winslow disponibilizou um *website* chamado *BioPortal*[1], que publica diversas ontologias biomédicas. Contudo, a estrutura

das ontologias disponíveis apresenta poucas semelhanças com a ontologia proposta em [Gonçalves et al. 2007]. Em [Tanantong et al. 2011] encontra-se uma ontologia para realizar inferências sobre tipos de arritmias com base em dados de ECG. Um fragmento da ontologia de [Gonçalves et al. 2007] é utilizado neste artigo.

Em [Ngo and Veeravalli 2014], os autores propõem uma plataforma baseada nas tecnologias da Web Semântica que permitem o armazenamento de parâmetros extraídos do sinal ECG em uma base de dados. Em [Trigo et al. 2012], Jesús Daniel Trigo et al. propõem o projeto e o desenvolvimento de um sistema para o gerenciamento interoperável de diferentes formatos de registros digitais de sinais ECG. Os autores propõem uma combinação de diferentes formatos de armazenamento de registros de ECG. Em [Khumrin and Chumpoo 2016], os autores propõem uma abordagem para a integração de dados eletrocardiográficos, originados de diferentes formatos, utilizando-se um formato de dados de referência baseado em classes Java.

3. A Abordagem LinkedECG

A Figura 1 ilustra as etapas da abordagem proposta (LinkedECG): extração de parâmetros, extração de dados e publicação de dados. Essas etapas serão descritas a seguir.

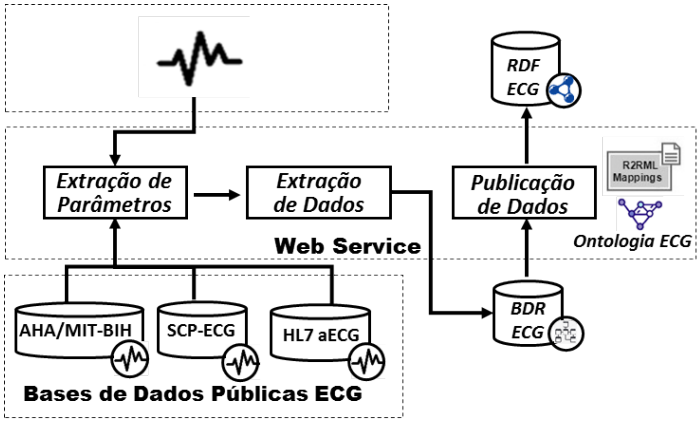


Figura 1. Visão Geral da Abordagem LinkedECG

3.1. Extração de Parâmetros (Features)

Esta etapa recebe como entrada um sinal ECG bruto, o qual pode ter sido originado de um biosensor ou de bases públicas previamente existentes (como por exemplo, AHA/MIT-BIH, SCP-ECG e HL7 aECG). Em seguida, um conjunto de algoritmos de processamento digital de sinais é executado, incluindo: filtragem para eliminação de ruído e interferências, aplicação de transformada *Wavelet* (análise tempo-frequência) para realce seletivo do complexo QRS e das ondas P e T, detecção dos picos e delineamento das formas de onda, extração de parâmetros (intervalos e amplitudes) [Madeiro et al. 2012]. Como resultado desta etapa, os seguintes parâmetros são extraídos: amplitude e duração de cada complexo QRS, intervalos entre batimentos, amplitude e duração de cada onda P, amplitude e duração de cada onda T, intervalos entre as diferentes formas de onda.

3.2. Extração de Dados

Esta etapa recebe como entrada uma matriz contendo os parâmetros extraídos do sinal ECG. Em seguida, esta matriz é processada e um conjunto de dados sobre o sinal ECG é

extraído e armazenado em banco de dados relacional, cujo esquema é ilustrado na Figura 2. A abordagem de armazenar os dados do sinal ECG em um banco relacional foi adotada com o objetivo de simplificar o processo de publicação dos dados, uma vez que já existem ferramentas que possibilitam a criação de *d*umps dos dados relacionais em formato RDF e a disponibilização desses *d*umps em *triplostore* RDF, de forma semiautomática.

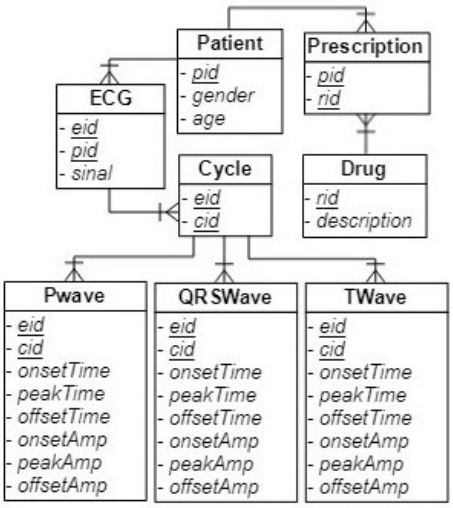


Figura 2. Esquema do banco de dados relacional.

3.3. Publicação de Dados

Nesta etapa, o banco relacional contendo as informações extraídas dos sinais ECG é acessado e os seus dados são exportados para o formato RDF. Dividimos esse processo em dois passos: No primeiro, cria-se um *d*ump dos dados relacionais em formato RDF. Para realizar este passo, utilizamos a ferramenta *D2RQ* [1] junto com os mapeamentos na linguagem R2ML [2], os quais relacionam o esquema do banco de dados relacional com o vocabulário da ontologia adotada para representar os sinais ECG. No segundo passo, os dados em formato RDF presentes no *d*ump gerado anteriormente são materializados em um *triplostore* RDF, mais especificamente no Virtuoso [3], de forma semiautomática. O Virtuoso disponibiliza um *SPARQL endpoint* que possibilita realizar consultas semânticas.

3.3.1. Adaptação de uma Ontologia de Sinais ECG

Inicialmente, buscamos utilizar uma ontologia de sinais ECG já existente. A ontologia proposta em [Gonçalves et al. 2007] foi a que mais se aproximou dos requisitos definidos para a abordagem LinkedECG. Desta forma, utilizamos os vocabulários $ecg :< http : //nemo.inf.ufes.br/biomedicine/ecg.html >$, disponível no trabalho de [Gonçalves et al. 2007], e $health :< https : //health - lifesci.schema.org/ >$ [4] (termos adicionais). O vocabulário $ecgo :< http : //www.arida.ufc.br/ecg >$ contém termos criados pela abordagem proposta. A Figura 3 apresenta a ontologia utilizada nesta etapa.

[1] http://d2rq.org/

[2] https://www.w3.org/TR/r2rml/

[3] https://virtuoso.openlinksw.com/rdf/

[4] Disponível em https://health-lifesci.schema.org/Patient

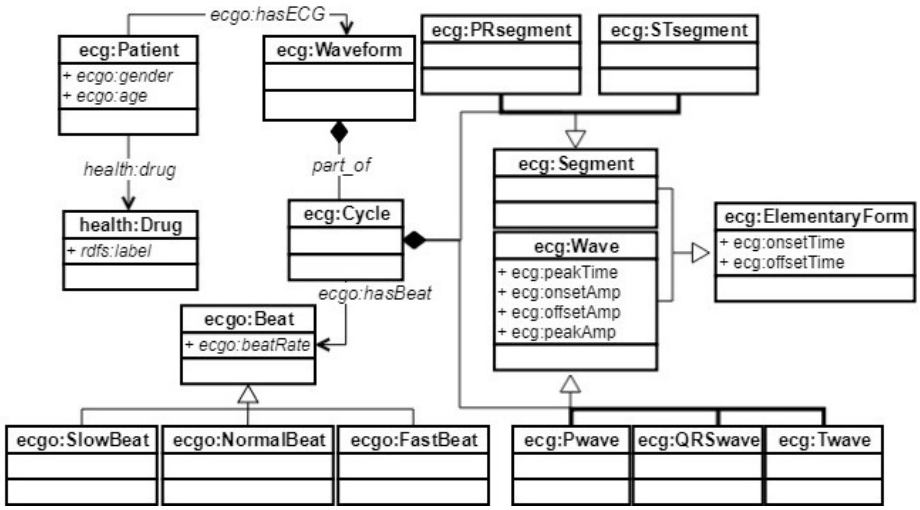


Figura 3. Ontologia utilizada na etapa de publicação dos dados.

4. Estudo de Caso

Realizar consultas sobre dados de sinais ECG é uma tarefa bastante complexa, uma vez que os dados são armazenados em arquivos de texto não estruturados. Por outro lado, a abordagem LinkedECG possibilita a execução de consultas semânticas. Para demonstrar este fato, apresentamos duas consultas que foram executadas sobre uma base de dados RDF gerada aplicando-se a abordagem LinkedECG.

Consulta Sparql 01: Quais pacientes do sexo masculino com mais de 60 anos tiveram algum batimento acelerado (acima de 100 batimentos por minuto) no ECG?

```
SELECT ?paciente
WHERE {?paciente a health:Patient ; health:age ?idade ;
            health:gerner ?sexo ; ecga:hasECG ?ecg .
       ?ciclo ecgo:part_of ?ecg .
       ?ciclo ecgo:hasBeat ?bat .
       ?bat a ecgo:FastBeat.
FILTER(?age >60 && ?sexo = "male")}
```

Figura 4. Consulta Sparql 01.

Consulta Sparql 02: Quais pacientes tomaram o medicamento *Aldomet* e apresentaram algum batimento cardíaco lento (abaixo de 60 batimentos por minuto) no ECG?

```
SELECT ?paciente
WHERE {?paciente a health:Patient ; health:drug ?med .
       ?med rdfs:label ?nomemed .
       ?paciente ecgo:hasECG ?ecg .
       ?ciclo ecgo:part_of ?ecg ; ecgo:hasBeat ?bat .
       ?bat a ecgo:SlowBeat.
FILTER (?nomemed = "Aldomet")}
```

Figura 5. Consulta Sparql 02.

5. Conclusões

Neste trabalho apresentamos uma abordagem, denominada LinkedECG, para a integração e publicação de dados de sinais ECG. A abordagem proposta possibilita a criação de uma

base de conhecimento pública, que pode ser usada para dar suporte a consultas complexas. Um estudo de caso foi realizado com a finalidade de comprovar que a LinkedECG facilita a execução de consultas que dificilmente seriam executadas sobre os arquivos contendo os sinais ECG brutos.

Como trabalhos futuros, iremos realizar testes de desempenho com a finalidade de avaliar o impacto proporcionado pela manipulação de dados RDF no tempo de resposta das consultas, bem como a escalabilidade da abordagem proposta. Além disso, iremos adicionar novas bases públicas de sinais ECG e explorar a utilização de algoritmos de mineração para a classificação/reconhecimento de arritmias e outros eventos adversos.

Referências

Bond, R. R., Finlay, D. D., Nugent, C. D., and Moore, G. (2011). A review of ECG storage formats. *International journal of medical informatics*, 80(10):681–697.

Geselowitz, D. B. (1989). On the theory of the electrocardiogram. *Proceedings of the IEEE*, 77(6):857–876.

Goldberger, A. L., Amaral, L. A., Glass, L., Hausdorff, J. M., Ivanov, P. C., Mark, R. G., Mietus, J. E., Moody, G. B., Peng, C.-K., and Stanley, H. E. (2000). Physiobank, physiotoolkit, and physionet. *Circulation*, 101(23):e215–e220.

Gonçalves, B., Guizzardi, G., and Pereira Filho, J. G. (2007). An electrocardiogram (ECG) domain ontology. In *Workshop on Ontologies and Metamodels for Software and Data Engineering, 2nd, João Pessoa, Brazil*, pages 68–81.

Heath, T. and Bizer, C. (2011). Linked data: Evolving the web into a global data space. *Synthesis lectures on the semantic web: theory and technology*, 1(1).

Khumrin, P. and Chumpoo, P. (2016). Implementation of integrated heterogeneous electronic electrocardiography data into maharaj nakorn chiang mai hospital information system. *Health informatics journal*, 22(1):34–45.

Madeiro, J. P., Cortez, P. C., Marques, J. A., Seisdedos, C. R., and Sobrinho, C. R. (2012). An innovative approach of QRS segmentation based on first-derivative, hilbert and wavelet transforms. *Medical engineering & physics*, 34(9):1236–1246.

Mandellos, G. J., Koukias, M. N., Styliadis, I. S., and Lymberopoulos, D. K. (2010). e-scp-ecg+ protocol: An expansion on SCP-ECG protocol for health telemonitoring—pilot implementation. *International journal of telemedicine and applications*, 2010:1.

Ngo, D. and Veeravalli, B. (2014). Applied semantic technologies in ecg interpretation and cardiovascular diagnosis. In *Bioinformatics and Biomedicine (BIBM), 2014 IEEE International Conference on*, pages 17–24. IEEE.

Tanantong, T., Nantajeewarawat, E., and Thiemjarus, S. (2011). Towards continuous electrocardiogram monitoring based on rules and ontologies. pages 327–330.

Trigo, J. D., Martínez, I., Alesanco, A., Kollmann, A., Escayola, J., Hayn, D., Schreier, G., and García, J. (2012). An integrated healthcare information system for end-to-end standardized exchange and homogeneous management of digital ECG formats. *IEEE Transactions on Information Technology in Biomedicine*, 16(4):518–529.

Investigando a Relação das Refatorações de Código com os Sentimentos de Mensagens de *Commit*

Jordão M. de Souza[1], Ticiana L. Coelho da Silva[1], Criston P. de Souza[1], Carla Ilane Moreira[1], Lincoln Rocha[1], José Antônio F. de Macêdo[1]

[1]Universidade Federal do Ceará (UFC)
Ceará – Brasil

jordao05@alu.ufc.br, jose.macedo@lia.ufc.br,

{ticianalc, criston, carlailane, lincolnrocha}@ufc.br

Abstract. *This paper presents an analysis of feelings in messages expressed by developers through commits in code repositories. In this work, six open source projects were analyzed, and a total of 12,113 commit messages. In the first phase, commit messages were classified into two categories (positive or negative). As a second phase, the paper investigates whether feelings are related to code refactoring activities, and concludes that when working with refactorings, the tendency is to express fewer negative feelings.*

Resumo. *Este trabalho apresenta um estudo a partir da análise de sentimentos em mensagens expressas pelos desenvolvedores por meio de commits em repositórios de código aberto. Foram analisados seis projetos de código aberto e um total de 12.113 mensagens de commit. Para o estudo foram classificadas as mensagens dos commits em duas categorias: positivo ou negativo. Em seguida, é realizada uma investigação se os sentimentos estão relacionados às atividades de refatoração de código. Como resultados deste estudo, tem-se indícios que quando se trabalha com refatorações a tendência é a expressar menos sentimentos negativos.*

1. Introdução

Métodos de análise de sentimentos foram inicialmente desenvolvidos para extrair a polaridade de sentimentos em textos curtos ou comentários em redes sociais onde existe grande interação do público, como em *tweets* [Thelwall et al. 2012]. Pesquisas recentes associam fatores humanos, como humor e emoções, com resolução de problemas [Graziotin et al. 2014], com linguagens de programação [Guzman et al. 2014] e com tarefas de refatoração de código [Singh and Singh 2017].

Estas pesquisas têm mostrado que as emoções afetam a qualidade do produto desenvolvido, a produtividade, a criatividade e a satisfação dos desenvolvedores [De Choudhury and Counts 2013]. Guzman *et al.* (2014) concluíram que *commits* feitos nas segundas-feiras e projetos desenvolvidos na linguagem de programação Java tendem a ter mais sentimentos negativos. Singh e Singh (2017) mostraram que, geralmente, os desenvolvedores expressam com mais frequência sentimentos negativos que positivos enquanto realizam a refatoração de código.

Neste trabalho, são analisados os sentimentos expressos pelos desenvolvedores em mensagens de *commits* em projetos de código fonte aberto. A questão a ser investigada neste trabalho é: "Quando se trabalha com refatoração de código, os sentimentos expressos tendem a ser positivos?". A forma como este trabalho abordou essa questão difere da forma abordada em [Singh and Singh 2017], pois aqui foram analisados todos os *commits* que tiveram refatorações de código, e não apenas uma amostra.

Seis projetos de código fonte aberto foram utilizados neste trabalho. Para verificar o relacionamento dos dados de refatoração de código, de não refatoração e os sentimentos coletados, são utilizados os seguintes métodos estatísticos para validar os resultados: *Student1s t-Test* [De Winter 2013] e *Wilcoxon Test* [Gehan 1965]. Como os *commits* com refatorações são disjuntos dos *commits* sem refatorações, os testes na tabela 4 foram feitos assumindo que as amostras são independentes (não pareadas). Foram encontradas evidências que mostram que quando se trabalha com tarefas de refatoração de código, os sentimentos negativos tendem a diminuir.

O restante do artigo é dividido como segue: a Seção 2 apresenta a metodologia para execução do estudo, indicando os projetos selecionados, como foi feita a análise dos sentimentos e a coleta de refatorações. A Seção 3 mostra os resultados encontrados e, por fim, a Seção 4 apresenta as conclusões e trabalhos futuros.

2. Metodologia do Estudo

Esta Seção apresenta a metodologia executada no estudo para investigação da relação das refatorações de código e sentimentos em projetos de código fonte aberto. É descrito o conjunto de dados analisados dos projetos extraídos do *Github*. Além disso, é apresentada como foram identificadas nos projetos as refatorações de código e realizada a análise dos sentimentos dos *commits* dos projetos.

2.1. Conjunto de Dados

Para realizar esse estudo, foram analisados projetos de código aberto codificados na linguagem Java extraídos do *GitHub*[1]. Os projetos foram escolhidos com base na quantidade de *commits* que continham. Foram selecionados os primeiros seis projetos que continham entre 4400 e 7850 *commits* dos *top-projects* na linguagem Java no GitHub. Os projetos analisados foram: *Dropwizard*, *Guava*, *Kafka*, *Mockito*, *RxJava* e *Tutorials*. A quantidade total de *commits* em cada projeto é ilustrada na Tabela 1. No entanto, nem todos os *commits* coletados foram analisados, como é explicado a seguir.

Para coleta das refatorações realizadas em cada um dos projetos, foi utilizada a ferramenta *RefactoringMiner* [Tsantalis et al.]. Para que essa ferramenta pudesse analisar as refatorações de um *commit*, este não poderia ser um *commit* de *merge*, pois este tipo de *commit* geralmente não tem alterações no código, apenas a junção de dois ou mais outros *commits*.

2.2. Análise de Sentimentos

Para analisar os sentimentos das mensagens de *commit* foi utilizada a ferramenta SentiStrength [Thelwall et al. 2012], que permite a extração de sentimentos a partir de textos

[1]https://github.com/

Projeto	Total de commits	Refatorações	
		Com	Sem
dropwizard	4482	101	951
guava	4676	346	1759
kafka	4871	392	3296
mockito	4653	516	1844
RxJava	5330	295	1542
tutorials	7817	147	924
Total	31829	1797	10316

Tabela 1. Quantidade total de *commits* por projeto

Score	Mensagem	Score Final
{4, -1}	Follow @fleaflicker's excellent[4]advice .	3
{3, -2}	Relax[3]the constraints[-2]on ConfiguredBundle .	1
{1, -5}	Definitely hating [-4][-1 booster word]#320 .	-4
{2, -2}	Correct default[-2]value[2] of rootPath	0

Tabela 2. Exemplos de mensagens de *commit*

curtos e de baixa qualidade. Esta ferramenta foi escolhida pela facilidade que tem de extrair sentimentos de textos curtos, e também porque já foi utilizada por estudos anteriores [Sinha et al. 2016, Singh and Singh 2017, Guzman et al. 2014].

SentiStrength utiliza um dicionário de *tokens*, que associa a cada *token* um *score*. Palavras com sentimentos negativos recebem uma pontuação entre -1 e -5, tal que -1 indica baixo sentimento negativo e -5 extremamente negativo. Já as palavras com sentimentos positivos recebem uma pontuação entre 1 e 5, sendo 1 baixo sentimento positivo e 5 extremamente positivo. Para analisar uma frase, a ferramenta atribui pontuações às palavras da frase que estão presentes no dicionário, e a pontuação final da frase é um par contendo o maior valor positivo e o maior valor absoluto negativo. Por exemplo, na frase "*I love you but I hate the current political climate*", a ferramenta associará à palavra "*love*" uma pontuação igual a 3 e a "*hate*", uma pontuação igual a -4. Logo, o *score* final da frase será {3, -4}. A Tabela 2 apresenta alguns *commits* dos projetos analisados e seus respectivos scores associados.

Para encontrar o sentimento final de uma mensagem do *commit*, foi realizada a soma da pontuação positiva e da pontuação negativa fornecida pela ferramenta *SentiStrength*, assim como é realizado em [Sinha et al. 2016]. Dessa forma, uma mensagem de *commit* pode ser classificada como: positiva quando o *score* final é maior que 0; negativa quando o *score* final é menor que 0; ou neutra quando o *score* final é igual a 0. Isso pode ser visto na Tabela 2. A coluna *Score final* mostra a pontuação final para cada exemplo.

Como a ferramenta SentiStrength é limitada ao seu dicionário e este deve ser específico para o contexto dos dados utilizados, isto pode ameaçar a qualidade da análise de sentimentos.

2.3. Refatorações

Segundo Fowler e Kent (1999) , refatoração de código é uma forma de mudar um sistema de software, melhorando a estrutura interna de uma forma que não mude seu comportamento externo. Ainda segundo eles, atividades de refatoração também minimizam os riscos de introdução a *bugs*.

A coleta das refatorações realizada pela ferramenta *RefactoringMiner* detecta 11 tipos de atividades de refatoração: *Extract Method*, *Inline Method*, *Move*

Tabela 3. Sentimentos entre *commits*

Sentimento	Score final do Sentimento	Com Refatoração		Sem Refatoração	
		Número de *Commits*	Porcentagem do Sentimento	Número de *Commits*	Porcentagem do Sentimento
Negativo	-4	0	39.73%	2	51.98%
	-3	7		36	
	-2	119		781	
	-1	588		4543	
Neutro	0	358	19.92%	1587	15.38%
Positivo	1	702	40.35%	3204	32.64%
	2	23		158	
	3	0		4	
	4	0		1	
Total	-	1797	-	10316	-

Method/Attribute, *Pull Up Method/Attribute*, *Push Down Method/Attribute*, *Extract Superclass/Interface*, *Move Class*, *Rename Class*, *Rename Method*, *Extract and Move Method* e *Change Package*). Mais detalhes sobre essas atividades de refatoração podem ser encontrados em [Tsantalis et al.]. *RefactoringMiner* implementa uma versão do algoritmo UMLDiff [Xing and Stroulia 2005] para modelos orientados a objetos. Esse algoritmo é usado para inferir o conjunto de classes, métodos e campos adicionados, excluídos ou movidos entre revisões de código sucessivas. Depois de executar esse algoritmo, um conjunto de regras é usado para identificar diferentes tipos de refatoração. A ferramenta *RefactoringMiner* pode ser utilizada independente de sistema operacional ou IDE. [Tsantalis et al. 2013] identificou uma precisão de 96,4% no uso da ferramenta, e mostra que há uma taxa muito baixa de falsos positivos.

3. Resultados e Discussões

Nas seções seguintes, é apresentada uma análise dos resultados encontrados.

3.1. Análise de Sentimentos e Refatoração de Código

Para encontrar a relação entre os sentimentos e as refatorações de código, foram analisadas separadamente as mensagens de *commits* com e sem refatorações. A quantidade de *commits* analisada para cada um dos 6 projetos é mostrada na Tabela 1, e a porcentagem de *commits* encontrada para os sentimentos negativos, positivos e neutros é mostrada na Tabela 3.

De modo similar ao procedimento de Sinha et al. (2016) , os *commits* com scores -1 e 1 (que resultariam em score final igual a 0) foram descartados. Desta forma, foi considerado como *commits* positivos aquele com score pelo menos 2, e *commits* negativos aqueles com score no máximo -2. Quando se trata de *commits* com refatorações, os sentimentos são balanceados entre negativos e positivos, com uma porcentagem de 39,7% e 40,4%, respectivamente. Já no restante dos *commits* (sem refatorações), o sentimento negativo domina, aparecendo em 52,0% dos *commits*, enquanto os sentimentos positivos apareciam em 32,6% das vezes. De fato, o Wilcoxon test [Gehan 1965] rejeitou a hipótese de que a distribuição dos scores dos *commits* sem refatoração é simétrica em relação ao zero (p-value $<$ 2,2e-16), embora também rejeite esta simetria para os *commits* com refatoração (p-value $<$ 0,019) se for levado em conta 95% de significância. Além disso,

Tabela 4. Testes por projetos nos *commits* com e sem Refatorações

	Com refatorações			Sem refatorações		
Projetos	**Média**	**Wilcoxon signed-rank test (p-value)**	**Student's T-Test (p-value)**	**Média**	**Wilcoxon signed-Rank test (p-value)**	**Student's T-Test (p-value)**
Dropwizard	-0.16	0.12	0.1	-0.33	**<2.2e-16**	**<2.2e-16**
Guava	-0.36	**8.291e-10**	**2.918e-10**	-0.38	**<2.2e-16**	**<2.2e-16**
Kafka	-0.47	**2.231e-15**	**<2.2e-16**	-0.6	**<2.2e-16**	**<2.2e-16**
Mockito	0.35	**<2.2e-16**	**<2.2e-16**	0.24	**<2.2e-16**	**<2.2e-16**
RxJava	0.23	**1.639e-05**	**1.183e-05**	0.04	0.13	0.13
Tutorials	-0.18	**0.03**	**0.03**	-0.24	**1.98e-12**	**9.825e-13**
Média	-0.06	**0.02**	**0.02**	-0.26	**<2.2e-16**	**<2.2e-16**

foi aplicado o *Student's t-Test* [De Winter 2013] para testar se o score médio dos commits positivos é igual ao score médio dos commits negativos, e com significância de 95% esta hipótese foi rejeitada nos dois caso (com e sem refatoração).

Concluiu-se que em ambos os casos, os sentimentos expressos pelos desenvolvedores tendem a ser negativos, porém quando se trabalha com atividades de refatoração, a tendência é de expressar menos sentimentos negativos do que quando não se trabalha com refatoração. Esses resultados diferem de Singh e Singh (2017), pois os autores encontraram apenas que quando se trabalha com refatorações de código, os desenvolvedores expressam mais sentimentos negativos que positivos, sem fazer uma comparação com o sentimento expresso nas atividades que não tiveram refatorações de código.

3.2. Análise de Sentimentos e Refatoração de Código por projeto

Os resultados encontrados para alguns projetos não foram os mesmos encontrado no geral. A Tabela 4 mostra os resultados de *p-value* encontrados nos testes e as médias para cada um dos projetos separadamente. Quando o *p-value* aparece em negrito, significa que a hipótese nula foi rejeitada (*p-value* <0.05).

Em relação aos *commits* com refatorações, no projeto *Dropwizard* não foi possível rejeitar as hipóteses nulas em ambos os testes. Nos outros casos as hipóteses nulas foram refutadas, porém nos projetos *Mockito* e *RxJava*, as médias são positivas, o que indica que nesses casos, os *commits* quando há refatorações tendem a ter sentimentos positivos. Observando os dois projetos na Tabela 4 de *commits* sem refatorações, pode-se verificar que as médias continuam positivas, porém menores que suas respectivas médias quando há refatorações. Dessa forma, pode-se concluir que nesses dois projetos os sentimentos expressos em geral são positivos, porém quando se trabalha com atividades de refatoração, os sentimentos tendem a ser mais positivos ainda. Para os demais casos, os resultados acompanham o resultado encontrado no geral.

4. Conclusões

Este trabalho apresentou um estudo de análise de sentimentos em mensagens de *commit* produzidas por desenvolvedores em projetos Java de código fonte aberto e sua relação com refatorações do código. Assim como em Sinha et al. (2016) , foram identificadas evidências que mostram que em geral, os sentimentos dos projetos tendem a ser negativos. No entanto, quando se trabalha com tarefas de refatorações, os *commits* tendem a

ser menos negativos. Também foram identificados casos de projetos onde a média dos sentimentos é positiva, essa média se torna ainda mais positiva quando se trabalhava com atividades de refatorações. Como trabalhos futuros, pretende-se coletar dados de mais projetos e relacionar os sentimentos de mensagens de *commit* com outros fatores, como a aparição de mau cheiro no código (ou do inglês, *bad smells*) [Fowler and Beck 1999].

Referências

De Choudhury, M. and Counts, S. (2013). Understanding affect in the workplace via social media. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work*, CSCW '13, pages 303–316, New York, NY, USA. ACM.

De Winter, J. C. (2013). Using the student's t-test with extremely small sample sizes. *Practical Assessment, Research & Evaluation*, 18(10).

Fowler, M. and Beck, K. (1999). *Refactoring: improving the design of existing code*. Addison-Wesley Professional.

Gehan, E. A. (1965). A generalized wilcoxon test for comparing arbitrarily singly-censored samples. *Biometrika*, 52(1-2):203–224.

Graziotin, D., Wang, X., and Abrahamsson, P. (2014). Happy software developers solve problems better: psychological measurements in empirical software engineering. *PeerJ*, 2:e289.

Guzman, E., Azócar, D., and Li, Y. (2014). Sentiment analysis of commit comments in github: An empirical study. In *Proceedings of the 11th Working Conference on Mining Software Repositories*, MSR 2014, pages 352–355, New York, NY, USA. ACM.

Singh, N. and Singh, P. (2017). How do code refactoring activities impact software developers sentiments? – an empirical investigation into github commits. In *2017 24th Asia-Pacific Software Engineering Conference (APSEC)*, pages 648–653. IEEE.

Sinha, V., Lazar, A., and Sharif, B. (2016). Analyzing developer sentiment in commit logs. In *Proceedings of the 13th International Conference on Mining Software Repositories*, pages 520–523. ACM.

Thelwall, M., Buckley, K., and Paltoglou, G. (2012). Sentiment strength detection for the social web. *Journal of the Association for Information Science and Technology*, 63(1):163–173.

Tsantalis, N., Guana, V., Stroulia, E., and Hindle, A. (2013). A multidimensional empirical study on refactoring activity. In *Proceedings of the 2013 Conference of the Center for Advanced Studies on Collaborative Research*, pages 132–146. IBM Corp.

Tsantalis, N., Mansouri, M., Eshkevari, L., Mazinanian, D., and Dig, D. Accurate and efficient refactoring detection in commit history. In *Proceedings of the 40th International Conference on Software Engineering*, ICSE 2018.

Xing, Z. and Stroulia, E. (2005). Umldiff: an algorithm for object-oriented design differencing. In *Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering*, pages 54–65. ACM.

Apoiando o processo de imputação com técnicas de aprendizado de máquina

Rodrigo Tavares de Souza[1], Rafael Castaneda Ribeiro[1], Claudia Ferlin[2], Ronaldo Ribeiro Goldschmidt[3], Luis Alfredo V. Carvalho[4], Jorge de Abreu Soares[1]

[1]Centro Federal de Educação Tecnológica Celso Suckow da Fonseca (CEFET/RJ)

[2]Pontifícia Universidade Católica do Rio de Janeiro (PUC/RJ)

[3]Instituto Militar de Engenharia (IME)

[4]Universidade Federal do Rio de Janeiro (UFRJ)

rodrigo.souza@eic.cefet-rj.br, rafael.ribeiro@cefet-rj.br, ferlin@inf.puc-rio.br, ronaldo.rgold@ime.eb.br, luisalfredo@medicina.ufrj.br, jorge@eic.cefet-rj.br

Abstract. *The task of imputation of missing data is an important challenge faced by data scientists. In this context, imputation techniques that improve the quality of the data entered are imperative. Exploring both machine learning techniques and variations of the classical imputation process can improve the quality of the imputed data. Hence, this article aims to evaluate the impact of the use of the k-neighbors algorithm faced to the use of the mean in the global imputation process as well as to explore the use of the hot-deck imputation technique with the clustering algorithm k-Means and imputation with k-NN. Results reveal an interesting reduction of absolute error obtained in the simulation in three databases with different characteristics.*

Resumo. *A tarefa de imputação de dados é um importante desafio enfrentado pelos cientistas de dados. Nesse contexto, torna-se imperativo dispor-se de técnicas de imputação que melhorem a qualidade do dado preenchido. Valer-se tanto de técnicas de aprendizado de máquina quanto de variações do processo clássico de imputação pode tornar possível a melhora da qualidade dos dados imputados. Assim, este artigo tem por propósito avaliar o impacto da utilização do algoritmo dos k-vizinhos mais próximos frente ao uso da média no processo de imputação global bem como explorar o uso da técnica de imputação hot-deck com o algoritmo de agrupamento k-Means e a imputação com k-NN. Os resultados revelam interessante redução da margem de erro obtida na simulação em três bases de dados com diferentes características.*

1. Introdução

O importante aumento da quantidade de dados gerenciados por sistemas informatizados é fator inegável na rotina das corporações. Todavia, esse crescimento maximiza um conhecido problema dos administradores de dados: as inconsistências de dados. Essas inconsistências são fruto primário dos diversos momentos onde bases de dados são integradas. Essas bases vêm de diversas fontes, que nem sempre recebem o devido cuidado. Podem também ocorrer por outros motivos, tais como falhas de equipamentos, na transmissão da mensagem, erros de preenchimento não tratados, falhas em rotinas de carga, entre outros [Han et al., 2011]. E um dos casos de inconsistência que demandam atenção é o da ausência de valores em bases de dados.

Essas ausências, dependendo de sua natureza e incidência, podem prejudicar sobremaneira a análise de dados por qualquer técnica produtora de informação, tais como a descoberta de conhecimento em bases de dados, os armazéns de dados (*Data Warehouse*), ou similares, e comprometer seus resultados [Little *et al*, 2002]. Para isso, o estudo de técnicas de complementação de dados ausentes (ou "Imputação") procura soluções para questões em aberto ligadas ao tema, fundamentalmente com base em métodos estatísticos e/ou técnicas de aprendizado de máquina [Farhangfar et al., 2007]. Assim, o objetivo desse trabalho é o de avaliar o impacto na qualidade do dado imputado (erro absoluto entre o que é calculado e o dado real) com técnicas de aprendizado de máquina, tanto na imputação global quanto local e, na ausência de uma técnica consagrada como estado da arte, frente à clássica imputação com o uso de média.

Vários trabalhos exploram a variação de técnicas de imputação, com vistas à melhoria da qualidade do dado imputado, tais como os de Luengo *et al* [2012] e Silva e Zárate [2014]. Porém, uma importante técnica com potencial de melhora da qualidade do dado imputado – ou seja, a diminuição do erro absoluto entre o valor real e o deduzido – é a imputação local ou *hot-deck* [Ford, 1983, Jerez et al., 2010]. Nesta técnica, divide-se o conjunto de dados em grupos, de forma que a composição do valor imputado se dê somente por elementos de tuplas que sejam similares aos elementos da tupla com valor ausente [Fuller et al., 2001]. Neste artigo, a imputação *hot-deck* é implementada por meio da combinação de dois algoritmos de aprendizado de máquina: o agrupamento utiliza o algoritmo *k-Means* [Han *et al*, 2011], e a imputação propriamente dita: o algoritmo dos *k*-vizinhos mais próximos (k-NN) [Han *et al*, 2011]. O desempenho desta abordagem é avaliado em três bases de dados com diferentes níveis de correlação entre seus atributos, variados percentuais de ausência em cada atributo de cada base, e múltiplos valores dos parâmetros dos algoritmos, como proposto por Soares [2007]. Espera-se com isso aproximar cada vez mais o dado imputado do dado real.

Além da introdução, o artigo está organizado da seguinte forma: a Seção 2 analisa a utilização de técnicas de aprendizado de máquina no processo de imputação global (que considera todos os valores presentes na coluna alvo da imputação) e local (que considera um subconjunto desses valores no processo de imputação). A Seção 3 explora os resultados com a utilização do algoritmo k-NN frente à média aritmética simples, e o uso da imputação local com o algoritmo de agrupamento *k-Means* e a imputação com o algoritmo dos *k* vizinhos mais próximos. Por fim, a Seção 4 tece as considerações finais do artigo.

2. Usando técnicas de aprendizado de máquina na imputação global e local

Existem diversos métodos para tratar a ausência de dados, ou seja, substituir valores ausentes por valores reais [Rubin,1988]. A tarefa de imputação tem como objetivo recuperar valores ausentes de maneira mais precisa, através de técnicas que variam desde a média simples, regressão linear, modelos preditivos específicos, até a utilização de algoritmos de aprendizado de máquina [Ford,1983, Jerez *et al*, 2010].

A maior parte dos trabalhos disponíveis na literatura realiza a tarefa de imputação, seja ela simples ou precedida de alguma outra técnica, com o cálculo da média aritmética simples dos valores presentes na coluna cujos valores são imputados (no caso de bases numéricas) ou com o uso da moda (em bases de dados categóricas). Apesar de simples, essa abordagem carrega consigo um considerável erro, por desconhecer qualquer similaridade das tuplas da base.

Nesse contexto, considera-se uma importante e difundida técnica de imputação que busca reduzir o desvio de similaridade entre os dados, classificando a priori a amostra, é denominada imputação local, ou *hot-deck* [Ford, 1983]. Nesta abordagem são utilizados somente grupos de objetos completos que possuam relação de similaridade com o dado ausente [Fuller *et al*, 2001]. O seu principal objetivo visa reduzir desvios

através da classificação da amostra [Ford, 1983], algo difícil de ser atingido [Soares, 2007].

Qualquer estudo envolvendo ausência de dados deve delimitar o mecanismo causador da ausência. Ausências de dados normalmente seguem um mecanismo de distribuição, mas também podem ocorrer de forma intermitente ou simplesmente ao acaso [Little, Rubin, 2002]. Trabalhos envolvendo a complementação de dados ausentes levam inevitavelmente em conta o mecanismo que causou a ausência dos dados, classificadas em três tipos [Little *et al*, 2002]: completamente aleatória – *Missing Completely at Random* (MCAR), aleatória – *Missing at Random* (MAR) e não aleatória – *Not Missing at Random* (NMAR).

3. Avaliação Experimental

Foram utilizadas três bases de dados numéricas com atributos classificadores no experimento, como proposto inicialmente por Soares [2007] e Castaneda *et al* [2008], disponíveis no repositório da Universidade da Califórnia, Irvine: *Iris Data Set*, *Breast Cancer Wisconsin (Original) Data Set*[1] e *Pima Indians Diabetes*[2] [Dua, Karra Taniskidou, 2017]. O conjunto de dados *Iris Data Set* relaciona as medidas de comprimento e largura das pétalas e caules de três espécies de plantas Iris. Já o dataset *Pima Indians* apresenta dados referentes a integrantes de uma tribo indígena, onde parte deles possui diabetes mellitus. Por fim, *Breast Cancer* possui dados do hospital de Wiscosin sobre o diagnóstico de câncer de mama, com dados relativos a pacientes que possuem esta doença. Os dados utilizados foram os originais, sem nenhum processo de normalização dos mesmos.

No que tange à simulação da ausência, adotou-se neste trabalho uma abordagem em largura: gerou-se ausência em todos os atributos de todas as bases, à exceção dos atributos identificadores e classificadores. Para cada base e atributo, provocou-se ausências de dados nas proporções de 10%, 20%, 30%, 40% e 50%, seguindo o mecanismo MCAR, conforme proposto por Soares [2007]. Limitou-se a ausência máxima em 50%, pois valores acima dessa taxa degeneram a base de modo que seu uso passa a ser questionável.

Adotou-se a mesma proposta ampla na variação do parâmetro *k* dos algoritmos de aprendizado de máquina aqui utilizados. O agrupamento precedendo a imputação é técnica conhecida e apresenta bons resultados [Little, Rubin, 2002]. Entretanto, existem poucos estudos que avaliam o número ideal de *k* vizinhos a serem considerados no algoritmo *k-NN* na utilização destes algoritmos no processo de imputação, bem como o número *k* de grupos do algoritmo *k-Means* [Soares, 2007]. Uma abordagem de variação dos valores do parâmetro *k* figura em Castaneda *et al* [2008], que utilizou os valores 1, 3, 5 e 10 para o algoritmo *k-NN* na composição de um *workflow* de imputação iterativa. De forma a explorar os possíveis valores de *k* para o *k-NN* e *k-Means*, foram utilizados todos os valores possíveis para esse parâmetro, nas três bases de dados, para os cinco percentuais de ausência [Soares, 2007]. A Tabela 1 detalha a configuração de *k* para os algoritmos, apresentando os valores mínimo e máximo do parâmetro para cada base, algoritmo e percentual de ausência.

[1] Disponíveis em http://archive.ics.uci.edu/ml/index.php.

[2] Disponível em https://www.kaggle.com/uciml/pima-indians-diabetes-database.

Tabela 1. Valores mínimo e máximo do parâmetro *k* para os algoritmos *k-Means* e *k-NN*

Base de dados	Algoritmo	Percentual de ausência				
		10	20	30	40	50
Iris Plants	k-NN	1-135	1-120	1-105	1-90	1-75
	k-Means	2-135	2-120	2-105	2-90	2-75
Pima Indians	k-NN	1-353	1-314	1-275	1-236	1-196
	k-Means	2-353	2-314	2-275	2-236	2-196
Breast Cancer	k-NN	1-614	1-546	1-478	1-410	1-341
	k-Means	2-614	2-546	2-478	2-410	2-341

As bases *Iris Plants* e *Breast Cancer* apresentam bons índices de correlação entre seus atributos e tendem a favorecer o processo de imputação [Soares, 2007]. Já a base *Pima Indians* apresenta baixo nível de correlação, sendo um dos desafios do experimento. Para aferir a atenuação geral (média) dos métodos em cada base, foram somadas as diferenças entre os erros dos mesmos, em cada percentual de ausência, sendo esse somatório divido por cinco (número de cenários de ausência), conforme especificado na Tabela 2.

Tabela 2. Total de correlações entre os atributos de cada base

Base de dados	Nº de correlações maiores que 50% / Nº de atributos	Correlação
Iris Plants	3 / 4	Alta
Pima Indians	3 / 8	Baixa
Breast Cancer	8 / 9	Alta

As Figuras 1, 2 e 3 mostram um comparativo médio de erro nos atributos de cada uma das bases com imputações feitas utilizando a média aritmética simples frente à complementação de dados ausentes realizada com o algoritmo *k-NN*. Os resultados na base *Iris Plants* são significativamente melhores, como por exemplo com 40% de ausência (erro médio de 75.75% e 9.32% para, respectivamente, os métodos de imputação média e k-NN).

Comportamentos interessantes são observados na base *Breast Cancer*. O uso do algoritmo *k-NN* frente à média revela ganhos consideravelmente animadores. No melhor caso, com 10% de ausência. Nesta situação, obteve-se um erro médio de 114.82% versus 37.46%. Além disso, tem-se que o erro oscilou na casa dos 37% para as diversas taxas de ausência, com desvio-padrão dos experimentos com a aplicação do método *k-NN* frente à média igual a 0.17%, frente a 1.46% na base *Iris Plants* e 2.27% na base *Pima Indians*. A Tabela 3 apresenta os desvios-padrão médios dos erros de imputação por base de dados e método de imputação. Esse é um resultado bem interessante, que demanda aprofundamento em bases com características similares. Isto é, um forte indicativo provavelmente reside no fato de que a alta correlação dos seus atributos ajuda sobremaneira o ganho.

Tabela 3. Desvios-padrão médios dos erros de imputação por base de dados e método de imputação

	k-NN			k-Means + k-NN		
	Iris Plants	Pima Indians	Breast Cancer	Iris Plants	Pima Indians	Breast Cancer
10%	5,85	33,04	37,46	5,59	31,86	36,59
20%	8,84	36,89	37,85	8,22	35,94	36,54
30%	7,59	39,09	37,61	7,29	34,88	37,13
40%	9,32	34,90	37,82	9,01	33,42	38,41
50%	9,20	36,59	37,79	9,12	36,19	38,40
Desvio-padrão	1,46	2,27	0,17	1,46	1,82	0,93

Já a base *Pima Indians*, cuja correlação entre os atributos é baixa, não se destaca como no caso anterior. Todavia, ainda assim os resultados são interessantes. Com 10% de ausência, temos erros médios de 43.15% para a média e 33.04% para o algoritmo dos *k* vizinhos mais próximos.

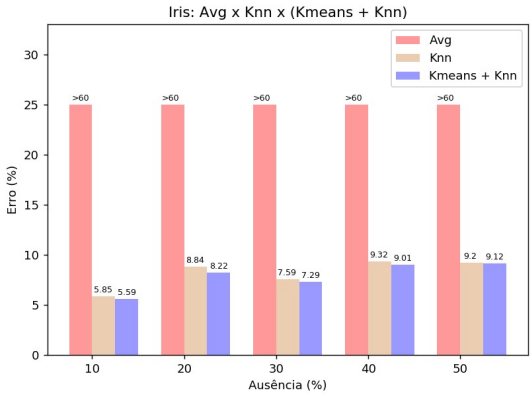


Figura 1. Erro médio na base *Iris Plants* com imputação por média, k-NN e hot-deck

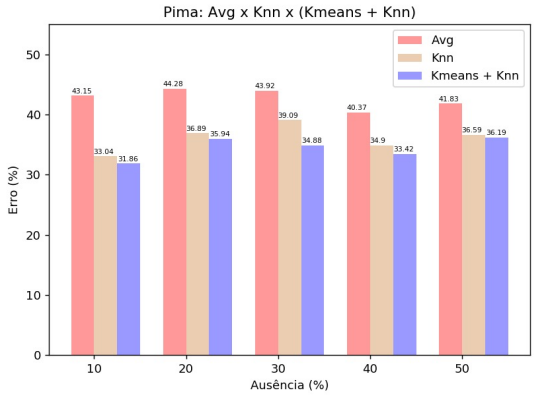


Figura 2. Erro médio na base *Pima Indians* com imputação por média, k-NN e hot-deck

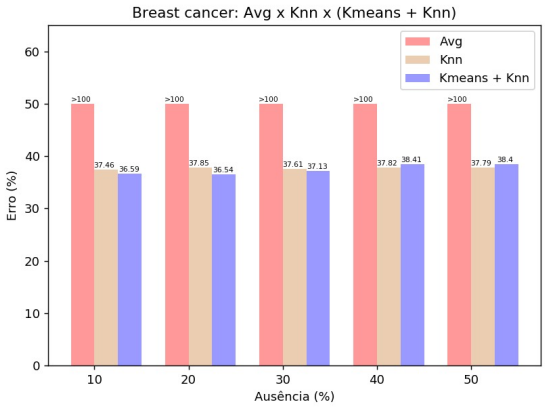


Figura 3. Erro médio na base *Breast Cancer* com imputação por média, k-NN e hot-deck

No que tange ao uso da imputação *hot-deck*, em praticamente todos os experimentos o erro médio de imputação diminui com o agrupamento precedendo a imputação. Apenas na base *Breast Cancer* essa situação se inverte com 40% e 50% de ausência. Todavia, os ganhos não são tão significativos quanto o são a utilização do algoritmo *k-NN* para imputação quando comparada ao uso da média para o mesmo objetivo, como pode ser observado na Tabela 4. Entretanto, os resultados também não desencorajam o investimento no tema, por ter de alguma forma contribuir com o ganho de uma técnica consolidada. As condições do estudo realizado neste artigo podem se revelar mais animadores em outro contexto.

Tabela 4. Diferença entre os erros médios entre a imputação com k-NN e *hot-deck* por percentual de ausência.

	Iris Plants			Pima Indians			Breast Cancer		
	k-NN	k-M + k-NN	Diff	k-NN	k-M + k-NN	Diff	k-NN	k-M + k-NN	Diff
10%	5,85	5,59	0,26	33,04	31,86	1,18	37,46	36,59	0,87
20%	8,84	8,22	0,62	36,89	35,94	0,95	37,85	36,54	1,31
30%	7,59	7,29	0,30	39,09	34,88	4,21	37,61	37,13	0,48
40%	9,32	9,01	0,31	34,90	33,42	1,48	37,82	38,41	-0,59
50%	9,20	9,12	0,08	36,59	36,19	0,40	37,79	38,40	-0,61

4. Conclusão

Neste artigo, experimentou-se o processo de imputação simples e local (*hot-deck*) utilizando duas consagradas técnicas de aprendizado de máquina: o algoritmo dos *k* vizinhos mais próximos para a imputação propriamente dita, e o agrupamento de dados com o algoritmo *k-Means* para a imputação local, além do uso clássico de imputação considerando apenas a média aritmética simples. Os resultados revelaram um promissor ganho de qualidade da imputação com o algoritmo *k-NN* frente ao uso da média aritmética simples. Ao adotar a técnica de imputação local, os resultados mostraram um ganho frente à imputação normal, mas com menos impacto. Esses resultados incentivam o investimento no tema, com a exploração de outras bases com diferentes características e com a utilização de outros algoritmos. Outro importante resultado revelou-se na imputação *hot-deck*, que teve o melhor resultado justamente onde apresenta os níveis de correlação mais desafiadores para o processo de imputação, a base *Pima Indians*. Utilizar a imputação *hot-deck* pode ser uma alternativa interessante para bases de dados que apresentem baixa correlação. Os próximos passos consistirão na aplicação de outras técnicas de aprendizado de máquina que extrapolem as tarefas de agrupamento, além da utilização de processamento de alto desempenho para lidar com bases de dados grandes.

Referências

Castaneda, R., Ferlin, C., Goldschmidt, R., Soares, J., Carvalho, L., Choren, R. (2008). Aprimorando Processos de Imputação Multivariada de Dados com Workflows. XXIII Simpósio Brasileiro de Banco de Dados (SBBD), pages 238–252.

Dua, D., Karra Taniskidou, E. (2017). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Farhangfar, A., Kurgan, L., Pedrycz, W. (2007). A novel framework for imputation of missing values in databases. IEEE Transactions on Systems, Man, and Cybernetics.

Ford, B. L. (1983). An Overview of Hot-Deck Procedures. Incomplete Data in Sample Surveys, 1 ed., vol. 2, Academic Press.

Fuller, W. A., Kim, J. K. (2001). Hot Deck Imputation for the Response Model. Survey Methodology, v. 31, n. 2, pp. 139-149.

Han, J., Kamber, M., Pei, J. (2011). Data Mining: Concepts and Techniques, 3ed. Morgan Kaufmann, Waltham, Mass.

Jerez, J. M., Molina, I., García-Laencina, P. J., Alba, E., Ribelles, N., Martín, M., Franco, L. (2010). Missing data imputation using statistical and machine learning methods in a real breast cancer problem. Artificial Intelligence in Medicine.

Little, R. J. and Rubin, D. B. (2002). Statistical Analysis with Missing Data. John Wiley & Sons,New York, 2ed.

Luengo, J., García, S., Herrera, F., (2012), On the choice of the best imputation methods for missing values considering three groups of classification methods, Knowledge and Information Systems, v. 32, n. 1 (Jul.), p. 77–108.

Rubin, D. B. (1988). An overview of multiple imputation. In Proceedings of the Survey Research Section, American Statistical Association, pp. 79–84.

Silva, L. O., Zárate, L. E. (2014). A brief review of the main approaches for treatment of missing data. Intelligent Data Analysis, vol. 18, no. 6, pp. 1177-1198.

Soares, J. (2007). Pré-processamento em Mineração de Dados: um Estudo Comparativo em Complementação. Tese de Doutorado, COPPE/UFRJ.

Uma Estratégia Eficiente de Treinamento para Programação Genética Aplicada a Deduplicação de Registros.

Davi Guimarães da Silva[1], Moisés Gomes de Carvalho[2], Duivilly Brito[2]

[1]Instituto Federal de Educação Ciência e Tecnologia do Pará (IFPA)

[2]Instituto de Computação - Universidade Federal do Amazonas (UFAM)

{davi.guimaraes}@ifpa.edu.br, {moises,db}@icomp.ufam.edu.br

Abstract. *Genetic Programming (GP) is a machine learning technique effectively used in the record deduplication problem. GP adopts a very expensive training step that requires that all records in a database be compared against each other several times. In this paper, we propose a novel approach for training step based on clustering technique combined with a sliding window. This combination aims at minimize the number of comparisons required in the training step without affecting its results. Our experiments using real datasets show that it is possible to reduce the time cost of the training step up to 72.8% compared to the GP state of the art approach without a significant impact in the quality of generated solutions.*

Resumo. *Programação Genética (PG) é uma técnica utilizada de forma eficaz na deduplicação de registros. Nela faz-se necessário realizar uma etapa de treinamento, em que cada registro é comparado com todos os outros na base de dados, tornando-a custosa. Neste artigo, propomos uma abordagem baseada na combinação de uma técnica de agrupamento e janela deslizante, visando minimizar a quantidade de comparações. Nossos experimentos com dados reais mostram que é possível reduzir o custo de treinamento da PG em até 72.8% comparado ao estado da arte sem uma redução significativa na qualidade das soluções geradas.*

1. Introdução

O problema de detecção e remoção de registros repetidos em um repositório é geralmente conhecido como deduplicação de registros [Koudas et al. 2006], que consiste em identificar e remover registros que são potencialmente os mesmos em uma base de dados.É uma tarefa complexa, que requer muito tempo e poder de processamento devido à grande quantidade de comparações necessárias para definir se um registro possui uma ou mais réplicas.

Em [Carvalho et al. 2008b], os autores apresentaram uma abordagem para a identificação de registros duplicados em repositórios recorrendo a uma técnica de Aprendizado de Máquina conhecida como Programação Genética (PG). Essa técnica apresenta alta precisão no processo de deduplicação em bases de dados com diferentes características. Porém, possui alto custo computacional da PG, tendo em vista que na etapa de treinamento, que visa "ensinar" a técnica identificar as características relevantes de boas soluções, cada registro é comparado com todos os outros registros, tornando pouco viável sua adoção.

Neste artigo propomos um método baseado na combinação de uma técnica de agrupamento e janela deslizante, para minimizar a quantidade de comparações exigidas na etapa de treinamento da PG. Os experimentos mostram que é possível tornar a etapa de treinamento da PG mais rápida mantendo o nível de qualidade das soluções, tendo em vista que técnica proposta obteve uma eficácia próxima da abordagem de [Carvalho et al. 2009] utilizado como *baseline*, com uma redução significativa no tempo de treinamento.

2. Trabalhos Relacionados

A tabela 1 apresenta abordagens utilizadas para o processo de deduplicação de registros com diferentes técnicas de aprendizagem de máquina.

Tabela 1. Abordagens aplicadas à deduplicação de registros.

Referência	Método Proposto
[Fellegi 1969]	Propôs um modelo baseado em probabilidades para otimizar a classificação dos pares de duas bases. Essa teoria é utilizada até os dias atuais
[Carvalho et al. 2008b]	Propõem uma abordagem baseada em PG que busca combinar evidências para a geração de funções de similaridade, utilizando uma pequena porção da base de dados para treino.
[Carvalho et al. 2008a]	Descrevem as principais propriedades da PG e apontam que a parametrização do algoritmo já proposto em [Carvalho et al. 2008b], pode alcançar resultados da deduplicação em até 30% melhores que anterior.
[Carvalho et al. 2009]]	Apresenta os resultados da PG aplicada a deduplicação de registro em três datasets: Cora, Restaurants e Synthetic obtendo melhor resultado que as abordagens anteriores.
[Carvalho et al. 2012]	Generalizam os resultados do método proposto, mostrando que a PG também é capaz de encontrar funções de deduplicação efetivas, mesmo quando as funções de similaridade não são conhecidas de antemão.
[Bianco; et al. 2013]	Propõem o arcabouço FS-Dedup para o processo de deduplicação de grandes volumes de dados, quando dependem de usuários especialistas para configurar as fases de blocagem e o algoritmo de classificação. Para isso, exploram algoritmos de deduplicação baseados em assinatura pela eficiência e escalabilidade.
[Ma et al. 2015]	Propõem uma nova nova abordagem para deduplicação de dados sem esquema e em paralelo utilizando MapReduce a partir de soluções que tornam o tamanho da janela deslizante adaptável, com a estratégia de múltiplas repetições com contagens adaptáveis visando acelerar o processo de deduplicação.
[Bianco; et al. 2016]	Os autores propõe uma estratégia de seleção de amostragem em duas fases, que vai desde a produção de pequenas subamostras aleatórias de pares candidatos em diferentes frações de conjuntos de dados e em seguida removem a redundância de pares para produzir um conjunto de treino menor e mais informativo.

É importante destacar que utilizamos a abordagem de [Carvalho et al. 2009] como *baseline*, por ser o mais recente com o uso da PG aplicada a deduplicação de registros e por isso os resultados são comparados com os deles. Além disso, os demais trabalhos da literatura utilizam outras técnicas para o mesmo problema.

3. Abordagem Proposta

A ideia desta proposta parte da abordagem de classificação que obedece as propriedades de transitividade das réplicas, em que, se um registro A for réplica de um registro B e B for réplica de um registro C, então A será réplica de C, e assim sucessivamente. Considerando essa transitividade, conclui-se que existe um nível de similaridade entre os registros réplicas disponíveis em um certo *dataset*. Assim, a nossa hipótese é que as réplicas terão distâncias equivalentes em relação ao um dado registro referência, a partir da similaridade entre cada registro, podendo diminuir a quantidade de comparação na fase de treinamento da PG.

3.1. Arquitetura geral da Abordagem Proposta

A Figura 1 apresenta uma visão geral da abordagem proposta com todas as etapas aplicadas à fase de treinamento da Programação Genética.

De modo geral, na **Etapa 1** ocorre a definição da estratégia de similaridade entre os registros de parte do *dataset*; na **Etapa 2** é realizada a comparação e ordenação dos registros em uma lista com base na similaridade retornada; na **Etapa 3** ocorre a definição dos parâmetros para a Janela Deslizante que será aplicada na lista ordenada; na **Etapa 4** é realizada a fase de treino da PG, e por fim os testes com parte do *dataset* não usada no treino para verificar quão boa foi a solução.

A abordagem proposta aplica os conceitos da técnica agrupamento baseada em particionamento de [Jain et al. 1999], em que os algoritmos de dividem os objetos entre os k *clusters* de acordo com a medida de similaridade adotada, de modo que cada objeto fique no

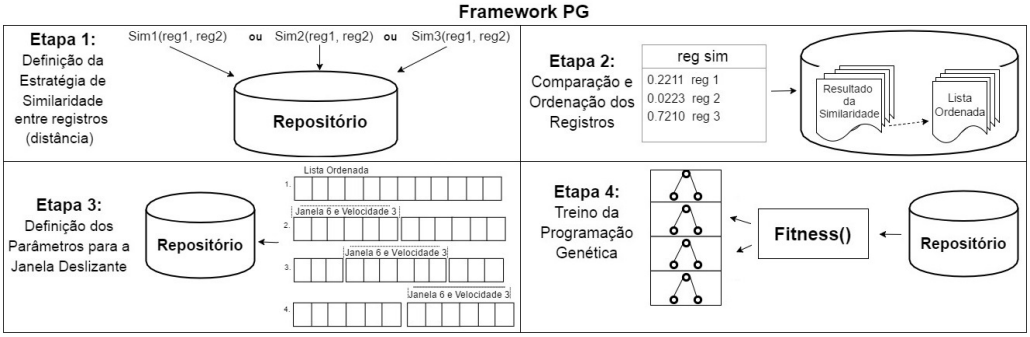


Figura 1. Visão geral da abordagem proposta.

cluster que forneça o menor valor de distância entre o objeto e o centro do mesmo. Assim, propomos os seguintes passos: 1) Escolher um registro *r* como o centro inicial do grupo; 2) Calcular a distância do registro *r* para cada um dos outros *n* registros utilizando uma função de similaridade; 3) Criar uma lista ordenada pelas distâncias retornadas pela função de similaridade; 4) Utilizar a abordagem de Janela Deslizante para comparar os registros que estiverem dentro do tamanho janela, obedecendo a lista ordenada anteriormente.

Tendo como base a classificação que obedece as propriedades de transitividade das réplicas, foram utilizadas três estratégias para escolha de um *registro referência*, isto é, o registro que será comparado com todos os outros do *dataset*, e que será usado com objetivo de definir a forma mais efetiva para realizar o agrupamento de registros similares.

Assim, foi testado o registro referência da seguinte forma: *1) Sintético:* É um registro criado a partir da junção entre atributos de registros diferentes existentes no *dataset* a ser avaliado. Esse registro deve conter todos os campos preenchidos. Desse modo, foi realizada a escolha de quatro registros aleatórios, visando a retirada de um atributo de cada para criar um novo registro. *2) Aleatório:* É escolhido um registro aleatoriamente dentre os registros existentes no *dataset* (sem ser previamente analisado). *3) String ruído:* Trata-se de um registro com as mesmas características do domínio dos tipos de dados existentes no *dataset*, contendo todos os campos preenchidos. O objetivo é representar o domínio dos tipos de atributo de uma determinada entidade, porém, não fica restrita aos valores do domínio, mas ao tipo de dados utilizado na representação daquele domínio.

Terminada essa etapa, os registros são ordenados por similaridade em ordem crescente. Para cada registro do *dataset* que possui réplica, é calculada uma distância desse registro para todas as sua(s) réplica(s). Assim, geramos as seguintes métricas: *Dmax*: Distância máxima encontrada entre um registro e qualquer uma de suas réplicas; *Dmed*: Distância média entre um registro e todas as suas réplicas. A partir desses valores, é calculada a média das distâncias máximas e a média das distâncias médias, para definir a função de similaridade que melhor agrupe um registro e suas réplicas, obtendo a menor *Dmax*. O objetivo é colocar os registros similares próximos uns dos outros.

3.2. Estratégia de Janela Deslizante

O próximo passo é comparar de todos os registros entre si. Para isso, foi adotada uma estratégia de janela deslizante proposta por [Ziv et al. 1977], na qual o autor propõe um modelo de "Janela Deslizante" de tamanho fixo ($w > 1$), que move-se sequencialmente sobre os registros ordenados a um deslocamento (v). Os registros que estiverem dentro da mesma janela serão comparados. Ao final, cada registro gerará ($2w$ - 1) pares para comparação, resultando um total de $O(nw)$ pares em um *dataset* com n registros no total.

A vantagem desta técnica é que a quantidade de pares comparados pode ser controlada e os registros que estiverem no intervalo definido pelo tamanho da janela, serão comparados. A partir da segunda execução os registros já comparados não serão mais comparados entre si, tendo impacto significativo no tempo de execução e reduzirá as comparações desnecessárias entre registros não similares.

3.3. Experimento para Definição da Função de Similaridade

A estratégia experimental de treinamento e testes utilizando PG adotadas neste trabalho, foram as mesmas do *baseline*, além das funções de similaridade e suas respectivas implementações: Softtfidf; Editdist; Jaro; Sortwinkler; Winkler; Bigrama; Bagdist; Seqmatch; e Compression. O objetivo foi encontrar uma função de similaridade que agrupasse registros similares mais próximos.

Em nossos experimentos utilizamos o *dataset Restaurants* [Bilenko 2003], que possui 864 registros divididos em 4 (quatro) *datasets* menores contendo 216 registros com o seguintes atributos: *(name, address, city, specialty)*. Na Tabela 2 são apresentados os resultados de *Dmax* e *Dmed* para cinco funções de similaridade que retornaram as menores distâncias no *dataset Restaurants*.

Tabela 2. Aplicação das funções de similaridade no *dataset Restaurants*.

Função	Distância *String* Ruido		Distância Registro Sintético		Distância Registro Aleatório	
	Dmax	Dmed	Dmax	Dmed	Dmax	Dmed
Editdist	**51**	**27**	62	29	52	31
Bagdist	61	35	64	36	61	34
Jaro	62	37	78	31	64	38
Sortwinkler	69	27	99	58	85	41
Bigram	72	33	155	46	75	36

Além dos resultados apresentados na tabela 2, também utilizaram-se as funções Softtfidf, Seqmatch, Winkler e Compression, que retornaram valores de *Dmax* e *Dmed* maiores para cada registro referência, porém *string ruido* sempre com distâncias menores. Assim, os experimentos mostraram que o uso do registro referência *string ruído* retorna os menores valores da *Dmax*, sendo aplicado para a avaliação do *dataset*.

3.4. Configuração Experimental para Janela Deslizante

Baseado nos resultados obtidos pela métrica *Dmed* no *dataset Restaurants* com o registro referência *string ruído*, foi definido para nossa experimentação utilizar janela deslizante inicialmente com aproximadamente metade do menor valor das *Dmed* obtidas no *dataset* e o tamanho do maior valor para janela um pouco acima do valor da média de todas as *Dmed* do *dataset*. Os valores do deslocamento foram baseados na porcentagem da quantidade de registros em cada *dataset*, ou seja, variaram em uma média de aproximadamente 3% a 10% em relação a quantidade de registros em cada *dataset*. Semelhantemente para os testes com as janelas utilizou-se três valores de deslocamento diferentes, descritos nas tabelas 3 e 4.

Para apresentar os resultados, criamos a Tabela 3 que exibe os resultados relacionados a Precisão, Revocação e F1 ([Baeza-Yates and Ribeiro-Neto 1999]), no treino e nos testes. A Tabela 4 apresenta os resultados do tempo gasto no treino (em segundos), a porcentagem do tempo em relação ao *baseline*, o total de comparações entre registros em cada configuração e a quantidade de réplicas identificadas corretamente. Para possibilitar a comparação da nossa abordagem com o *baseline*, criamos um campo descrito como ***"PG***

configuração padrão" no qual são apresentados os resultados obtidos, tanto para treino quanto para teste, tendo em vista que os experimentos reportados pelos autores não apresentaram os tempos de execução. Assim, foram executados nas mesmas condições de *hardware*, *software* e *dataset*, e serão descritos nas tabelas 3 e 4 a seguir.

Tabela 3. Resultados dos experimentos no *dataset Restaurants*

Configurações Gerais		Treino			Teste		
Parâmetros		Precisão	Revocação	F1 (σ)	Precisão	Revocação	F1 (σ)
PG configuração padrão		1.000	1.000	1.000 $\pm$(0.000)	1.000	0.963	0.981$\pm$(0.019)
Tam. janela	Deslocamento						
12	7	1.000	0.570	0.726 $\pm$(0.218)	1.000	0.535	0.694$\pm$(0.236)
23	7	1.000	0.801	0.889 $\pm$(0.100)	1.000	0.800	0.826$\pm$(0.109)
41	7	1.000	0.779	0.876 $\pm$(0.111)	1.000	0.713	0.858$\pm$(0.144)
	15	1.000	0.775	0.873 $\pm$(0.113)	1.000	0.750	0.856$\pm$(0.125)
58	7	1.000	0.953	0.976 $\pm$(0.024)	1.000	0.930	0.949$\pm$(0.036)
	15	1.000	0.810	0.895 $\pm$(0.095)	1.000	0.792	0.871$\pm$(0.105)
71	7	1.000	0.997	0.988 $\pm$(0.012)	1.000	0.949	0.975$\pm$(0.026)
	15	1.000	0.868	0.929 $\pm$(0.066)	1.000	0.829	0.898$\pm$(0.086)
	23	1.000	0.829	0.907 $\pm$(0.086)	1.000	0.802	0.889$\pm$(0.099)

Destaque-se que a precisão foi 1.0, isso significa apesar da redução na quantidade exemplos de treino, o método aprendeu com os exemplos existentes na janela dada. A tendência observada é que os valores de F1 melhoram com o aumento do tamanho da janela e a diminuição do valor de deslocamento desta. Ao realizar teste estatístico T com intervalo de confiança 95%, podemos identificar que muitas configurações de janela/deslocamento atingem desempenho de F1 estatisticamente equivalente à configuração padrão.

A Tabela 4 apresenta os resultados do tempo gasto na etapa de treinamento, a porcentagem comparada a *PG configuração padrão* e a quantidade de réplicas encontradas.

Tabela 4. Resultados dos experimentos no *dataset Restaurants*

Configurações (*Dataset* com 216 Registros)		Tempo do Treino (Segundos)		Total de Comparações	Réplicas Identificadas
(Total de réplicas no *dataset*: 10)		Total (seg)	%Tempo	Qtd (unid)	Qtd (unid)
PG configuração padrão		4466850	100%	23220	10
Tam. janela	Deslocamento				
12	7	14994	3.3%	1695	7
23	7	341260	7.6%	3860	8
41	7	68190	15.3%	7295	8
	15	592770	13.3%	6291	8
58	7	1095330	24.5%	10020	9
	15	881510	19.7%	9196	8
71	7	1214322	27.2%	11929	10
	15	1177405	26.4%	11046	9
	23	953950	21.4%	10675	9

Observando o resultado do tempo com a janela de tamanho 58 e deslocamento 7, o processo foi realizado em 24.5% do tempo comparado ao *PG configuração padrão*, encontrando 9 das 10 réplicas. Já a janela com tamanho 71 e deslocamento 7 realizou com 27.2% do tempo e encontrou todas as réplicas existentes. Houve uma redução de 72.8% do tempo gasto pela abordagem de [Carvalho et al. 2009] no treino da PG.

A partir dos resultados apresentados conclui-se que a tendência é que quanto maior o tamanho da janela e menor o deslocamento, os resultados são melhores. Quanto maior a janela, mais exemplos são utilizados no treinamento da PG. Já em relação ao deslizamento da janela, quanto maior o tamanho do deslocamento, as réplicas que poderiam estar em um determinado intervalo tendem a ficar em janelas diferentes e não serão comparados.

4. Conclusões e Trabalhos Futuros

Este trabalho propôs um método baseado na combinação de uma técnica de agrupamento e janela deslizante para a etapa de treinamento da PG, em que a partir da seleção de um registro referência, todos os outros registros foram comparados a ele por uma função de similaridade pré estabelecida, normalmente específica por base. Com os registros já agrupados com base nas distâncias retornadas pela similaridade, os mesmo foram comparados por uma estratégia de janela deslizante. A nossa combinação das técnicas, foi testada e comparada com os resultados do *baseline* e reforçaram a ideia de que a utilização dessa técnica permite a obtenção de resultados satisfatórios, com a vantagem da redução considerável no tempo de treinamento da PG, mantendo a qualidade dos resultados. Dessa forma, o diferencial desta proposta para o *baseline* é que sua aplicação foi específica para fase de treinamento da PG. Para trabalhos futuros planejamos realizar experimentos combinando as função de comparação entre os registros, utilizar paralelismo para a utilização da estratégia de janela deslizante, testar outros *datasets* de dados reais com diferentes domínios e graus de dificuldade, para ajudar a consolidar e estender os resultados obtidos neste trabalho.

Referências

Baeza-Yates, R. A. and Ribeiro-Neto, B. A. (1999). Modern Information Retrieval. *ACM Press/Addison-Wesley.*, New York, NY, USA. p. 39-48. (KDD 03).

Bianco;, G. D. et al. (2013). Tuning large scale deduplication with reduced effort. *In Proceedings International Conference on Scientific and Statistical Database Management, ACM, new york.*, p. 18:1-18:12. (SSDBM).

Bianco;, G. D. et al. (2016). A practical and effective sampling selection strategy for large scale deduplication. *IEEE International Conference on Data Engineering*, p. 1518-1519.

Bilenko, M.; Mooney, R. J. (2003). Adaptive Duplicate Detection Using Learnable String Similarity Measures. *In: ACM.*, New York, NY, USA. p. 39-48. (KDD 03).

Carvalho, M. G. et al. (2008a). The impact of parameter setup on a genetic programming approach to record deduplication. *S.B.C.; Brazilian Symp. Databases*, p.91-105.

Carvalho, M. G. et al. (2008b). Replica identification using genetic programming. *In: ACM Symposium on Applied Computing.*, p. 1801-1806.

Carvalho, M. G. et al. (2009). Evolutionary approaches to data integration related problems. *Tese. Universidade Federal de Minas Gerais*, p. 66-81.

Carvalho, M. G. et al. (2012). A genetic programming approach to record deduplication. *IEEE Transactions on Knowledge and Data Engineering; NJ, USA.*, v.24, p. 399-412.

Fellegi, I. P; Sunter, A. B. (1969). A theory for record linkage. *Journal of the American Statistical Association.*, [S.l.], v.64, n.328, p. 1183-1210.

Jain, A. K. et al. (1999). Data clustering: A review. *ACM Computing Surveys. 31(3)8.*

Koudas, N. et al. (2006). Record linkage: Similarity measures and algorithms. *ACM International Conference on Management of Data.*, p. 802-803, Chicago, USA.

Ma, K. et al. (2015). Large-scale schema-free data deduplication approach with adaptive sliding window using mapreduce. *The Computer Journal*, 58, n. 11, p.3187-3201.

Ziv, J. et al. (1977). A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, 23(3) pp. 337-343.

Detecção de Anomalias Frequentes no Transporte Rodoviário Urbano*

Ana Beatriz Cruz[1], João Ferreira[1], Diego Carvalho[1], Eduardo Mendes[4] Esther Pacitti[3], Rafaelli Coutinho[1], Fabio Porto[2], Eduardo Ogasawara[1]

[1] CEFET/RJ

[2]LNCC - DEXL Lab

[3]FGV

[4]Inria & University of Montpellier

anacruz@acm.org, joao.parana@acm.org, d.carvalho@ieee.org,

eduardo.mendes@fgv.br, renato.souza@fgv.br, Esther.Pacitti@inria.fr,

rafaelli.coutinho@cefet-rj.br, fporto@lncc.br, eogasawara@ieee.org

Abstract. *The growth of urban population and, consequently, the number of vehicles causes the increase of traffic jams and emission of polluting gases. In this context, we observe the intensification of papers that aim to identify bottlenecks and their causes. These papers propose methodologies that use trajectory data model and aim to explain systemic behaviors. This article proposes the identification and classification of anomalies in the urban road transport system from space-time aggregations to permanent objects. The methodology consists of pre-processing of data, identification of anomalies, identification, and classification of frequent patterns. Through it, we can identify the systemic and specific behaviors on the urban transit of Rio de Janeiro.*

Resumo. *O crescimento da população urbana e, consequentemente, do número de veículos provoca o aumento de engarrafamentos e da emissão de gases poluentes. Nesse contexto, observa-se a intensificação de pesquisas que buscam identificar engarrafamentos e suas causas. Estas pesquisas propõem metodologias que usam modelo de dados de trajetória e visam explicar comportamentos sistêmicos. Este artigo propõe a identificação e a classificação de anomalias no sistema de transporte rodoviário urbano a partir de agregações espaço-temporais a objetos permanentes. A metodologia consiste do pré-processamento dos dados, identificação de anomalias, identificação e classificação de padrões frequentes. Por meio dela, é possível identificar comportamentos sistêmicos e pontuais do trânsito urbano do Rio de Janeiro.*

1. Introdução

Em 2007, pela primeira vez existiam mais pessoas vivendo em áreas urbanas do que em zonas rurais, resultado de uma urbanização expressiva que se impulsionou desde a década

*Os autores agradecem à FAPERJ, à CAPES e ao CNPq pelo financiamento parcial do projeto.

de 1950 [United Nations, 2014]. Atualmente, mais da metade da população mundial vive em áreas urbanas exigindo um replanejamento dos serviços públicos das zonas urbanas para provê-los de maneira sustentável e duradoura. Outrossim, os desafios relacionados ao desenvolvimento eficiente e sustentável dos serviços de transporte passaram a ser investigados [Chen et al., 2015], levando à uma intensificação das pesquisas que conjugam a análise de dados de transporte à mobilidade urbana, com o objetivo de se identificar fenômenos causadores dos estrangulamentos do transporte, como os engarrafamentos.

Os estudos sobre os estrangulamentos fazem análise de dados coletados principalmente a partir de sistemas GP/dispositivos móveis embarcados em veículos que participam do fluxo, como táxis [Ferreira et al., 2013] e ônibus [Bierlaire et al., 2013]. Os dados coletados são frequentemente modelados como trajetórias de objetos móveis, pontos de início ou fim de movimento. Para análises mais sistêmicas, métodos de agregação espaço-temporal são usados para associar as observações às posições geográficas predefinidas (objetos permanentes) e esse tipo de agregação reduz significativamente o volume de dados de trajetórias [Tao et al., 2004].

Neste contexto, observa-se a necessidade de um estudo mais aprofundado sobre ótica das séries espaço-temporais de objetos permanentes que possam trazer uma melhor compreensão do tráfego [Cruz et al., 2017]. Este trabalho tem por objetivo identificar e classificar anomalias em dados agregados de mobilidade urbana. Para isso, uma técnica de identificação de anomalias nos comportamentos do trânsito é aplicada sobre os dados agregados por regiões predefinidas. Para extrair conhecimentos destas anomalias, um novo método de classificação de padrões frequentes é proposto, permitindo identificar padrões anômalos inesperados e esperados.

Além dessa introdução, o trabalho se divide em quatro outras seções. A seção 2 apresenta conceitos essenciais para o entendimento do problema e da metodologia adotada. Na seção 3, apresenta-se a metodologia aplicada. A seção 4 descreve uma avaliação da proposta. Finalmente, a seção 5 apresenta as conclusões e os próximos passos.

2. Fundamentação Teórica

As séries espaço-temporais são definidas como sequências de observações de objetos que contêm dados sobre o local e momento das coletas [Cressie and Wikle, 2015]. As observações podem ser emitidas por objetos permanentes ou móveis. Os objetos permanentes possuem localização fixa (sensores fixos) e os objetos móveis apresentam localizações que variam com o tempo (trajetória). O modelo de dados espaço-temporal mais aplicado a problemas relacionados ao tráfego é o de trajetória [Chen et al., 2015]. Os dados emitidos por sensores de posicionamento, como o GPS, possuem informações de latitude, de longitude e do momento da coleta. Desta forma, a sequência de dados coletados por sensores de posicionamento configura naturalmente uma trajetória, sem a necessidade de pré-processamento. Entretanto, estudos que se relacionam mais diretamente com o tema deste trabalho são aqueles em que agregam informações do objeto [Tao et al., 2004], gerando séries espaço-temporais associadas a objetos permanentes.

A partir de observações sobre um sistema é possível encontrar características específicas. A recorrência dessas características torna-as esperadas. Desvios significativos nas propriedades das características esperadas do sistema são consideradas anomalias [Aggarwal, 2016]. No contexto de mobilidade urbana, anomalias podem indicar

mudanças no comportamento por engarrafamentos ou aumento da velocidade. Elas podem ser causadas por acidentes, eventos, obras, operações de controle como *blitz* e Lei Seca, protestos, desastres e feriados. São difíceis de serem encontradas e interpretadas devido ao grande volume de dados e ao grande número de ruídos que compõem o conjunto de dados analisado [Lakhina et al., 2004]. Dessa forma, a identificação de anomalias é frequentemente feita sobre dados de trajetória.

A mineração de padrões frequentes é um dos métodos para identificar padrões que ocorrem com frequência no conjunto de dados analisado Han et al. [2011]. Esses padrões podem ser itens, sequências ou estruturas. Em todos os casos, por meio da mineração de padrões frequentes são identificadas regras de associação e correlações. Uma regra de associação é uma implicação de formato $X \rightarrow Y$. Seja $I = \{i_1, i_2, ..., i_n\}$ um conjunto de todos os itens, o antecedente da implicação (X) e o consequente (Y) são conjuntos de itens em I no qual nenhum item em X pertencente a Y e vice-versa. Logo, $X \cap Y = \emptyset$. A partir das regras de associação são identificados itens que ocorrem com frequência em uma mesma transação. Para que regras de associação sejam consideradas importantes, condições para suporte, confiança e correlação *lift* geralmente devem ser satisfeitas.

Entre as técnicas de mineração de padrões frequentes mais difundidas, destaca-se o algoritmo Apriori [Han et al., 2011]. Ele se baseia no princípio de que um conjunto de itens será frequente se todos os seus subconjuntos também forem. Em mobilidade urbana, algoritmos de identificação de padrões sequenciais frequentes são geralmente aplicados como base para identificar padrões frequentes em trajetórias [Giannotti et al., 2007].

Diversas medidas de classificação de padrões frequentes já foram propostas e elas se dividem em objetivas e subjetivas. As técnicas objetivas baseiam-se em propriedades estatísticas e as técnicas subjetivas, em conhecimentos de especialistas sobre o domínio [Mcgarry, 2005]. Os valores de suporte, confiança e *lift* são obtidos a partir de cálculos estatísticos e são classificadas como objetivas. As técnicas objetivas frequentemente retornam algumas regras que já são conhecidas ou triviais. Por outro lado, apesar da eficácia de técnicas subjetivas e sua influência na qualidade da classificação de padrões, a técnica é custosa devido à dificuldade e à complexidade para aquisição de conhecimentos prévios. Nesse contexto, pesquisas tem sido desenvolvidas aplicando-se uma combinação de técnicas subjetivas e objetivas a fim de extrair as vantagens de cada uma.

3. Metodologia

Este trabalho tem como objetivo identificar e classificar anomalias que ocorrem no sistema de transporte rodoviário urbano a partir das séries espaço-temporais derivadas de agregações das geolocalizações emitidas pelos ônibus da cidade do Rio de Janeiro[1]. Para isso, o processo é dividido em cinco etapas: pré-processamento (1) *Extract, Transform and Load* (ETL) e (2) agregação espaço-temporal; (3) identificação de anomalias; (4) identificação de padrões frequentes e (5) avaliação de padrões frequentes.

No pré-processamento (etapa 1), os dados são tratados até que possam ser minerados. Duas etapas principais o compõem: ETL e agregação espaço-temporal. A etapa ETL é responsável pela extração e limpeza dos dados abertos de mobilidade da cidade do Rio de Janeiro. Por meio da agregação espaço-temporal (etapa 2), os dados são convertidos

[1]Os dados são obtidos pelo Portal de Dados Abertos da Prefeitura do Rio de Janeiro: http://data.rio/

em séries espaço-temporais associadas a objetos permanentes, resultando em uma visão alternativa dos dados de mobilidade. A etapa seguinte (etapa 3) consiste na identificação de anomalias para compreensão de impactos de eventos, mudanças no tráfego e feriados. Neste presente trabalho, as anomalias nas séries espaço-temporais são identificadas com a aplicação de conceitos estatísticos conforme proposto em Cruz et al. [2017].

De modo a extrair conhecimento útil, as anomalias são estudadas por meio da aplicação da técnica de identificação de padrões frequentes. Os padrões identificados, chamados de regras de associação, configuram uma implicação lógica, na qual o antecedente é chamado condição e o consequente é chamado consequência. O algoritmo de identificação de padrões frequentes adotado (etapa 4) é o Apriori. O algoritmo de identificação de padrões frequentes é executado sobre as anomalias do transporte rodoviário variando-se os valores de suporte e confiança. Para cada valor de cada parâmetro selecionado, são avaliados os números de regras de associação resultantes e a qualidade dessas regras. A identificação de padrões frequentes é aplicada a duas granularidades diferentes: anomalias identificadas ao longo do ano e em cada mês individualmente.

Para classificar os padrões frequentes identificados, um método adaptado da técnica proposta por Liu et al. [2000], que combina abordagens subjetivas e objetivas de classificação, é proposto (etapa 5). Em Liu et al. [2000], a seleção de regras esperadas é feita por meio da informação dos conhecimentos de especialistas. De modo a não depender dos conhecimentos de um especialista, a primeira etapa da técnica de classificação de padrões proposta consiste em encontrar padrões que ocorram em uma visão anual das anomalias. Dessa forma, aplica-se o algoritmo de identificação de padrões frequentes sobre as anomalias identificadas ao longo de um ano. As regras de associação produzidas na visão anual são consideradas regras esperadas se possuem apenas um termo na condição e se o suporte e o *lift* são superiores aos valores mínimos passados como parâmetro.

A classificação de padrões frequentes é, então, aplicada a menores granularidades (em cada mês) segundo os valores de *lift* e de regras não esperadas (RNE), que mensura o quão não esperada é uma regra. Para calcular o valor de RNE, as condições (antecedentes) e consequências das regras são avaliadas, recebendo valor entre 0 e 1 para sinalizar se o antecedente (AE) e a consequência (CE) são esperados. Para isso, os termos que compõem as condições e consequências das regras produzidas por meio da identificação de padrões frequentes de menor granularidade são comparadas com os termos das regras esperadas produzidas na visão anual para gerar os valores AE e CE. O valor de mensuração de regras esperadas (RE) é resultante da média aritmética dos valores AE e CE. O valor de RNE é o inverso do valor de RE. Assim, quanto maiores são os valores de RNE e do *lift*, maior será a relevância da regra.

4. Avaliação Experimental

Essa seção descreve os resultados obtidos pela abordagem proposta sobre dados coletados de maio de 2014 a abril de 2015. No ano de 2014, observaram-se medidas que visavam melhorias na dinâmica do trânsito no Rio de Janeiro e o Brasil sediou a Copa do Mundo. A metodologia de identificação de anomalias aplicada sobre período avaliado resultou na identificação de 398.662 anomalias. Com exceções de julho e de agosto de 2014, o número de anomalias identificadas em todos os meses é inferior à 1% do volume de dados total de cada mês. De modo geral, domingos possuem mais anomalias

de velocidade maior que o intervalo típico em todos os meses. O mês de maio de 2014 teve maior número de anomalias que indicam velocidades inferiores ao intervalo típico. Nesse período, ocorreram diversas manifestações, greves de rodoviários e muitas obras na cidade do Rio de Janeiro ainda não haviam sido finalizadas.

A técnica de alisamento por média foi aplicada nos dados de horário e velocidade para discretizá-los e cada unidade espacial foi associada ao bairro no qual está localizado. Foram usados o suporte mínimo de 1% e a confiança mínima de 39% no algoritmo *Apriori* para identificação de padrões. O *Apriori* foi executado para anomalias identificadas ao longo do ano e em cada mês individualmente. Foram produzidas 292 e 2614 regras de associação para as perspectivas anual e mensal, respectivamente.

Em problemas de padrões frequentes, espera-se encontrar regras inesperadas, novas ou que tenham utilidade em tomadas de decisões e ações de especialistas. Algumas regras de associação produzidas não agregam conhecimentos relevantes. Por isso, aplica-se a metodologia de classificação das regras. As regras esperadas apresentaram *lift* > 1 e suporte relativamente alto. Por esse motivo, regras com apenas um termo no antecedente, *lift* ≥ 1 e suporte $> 0,05$ foram classificadas como regras esperadas. Foram identificadas 23 regras esperadas. Por exemplo, a regra $\{velocidade = Lenta\} => \{tipo = Menor\}$ é esperada. Ela indica que quando a velocidade é *Lenta*, a anomalia é do tipo *Menor*, ou seja, inferior ao intervalo típico. Elas servem de base para classificar as regras produzidas com a aplicação do *Apriori* sobre anomalias do ano e de cada mês individualmente.

A Figura 1 ilustra o número de regras por classificação sob perspectiva de mês e ano. A análise sobre as regras de associação para anomalias identificadas ao longo do ano identificou 70 regras como inesperadas. Aproximadamente 70% das regras são consideradas esperadas. A mesma análise foi feita para anomalias identificadas a cada mês individualmente. Para esta perspectiva, apenas 583 regras foram consideradas completamente inesperadas, ou seja, $RNE = 1$, e 1.282 regras, onde $0 <$RNE< 1.

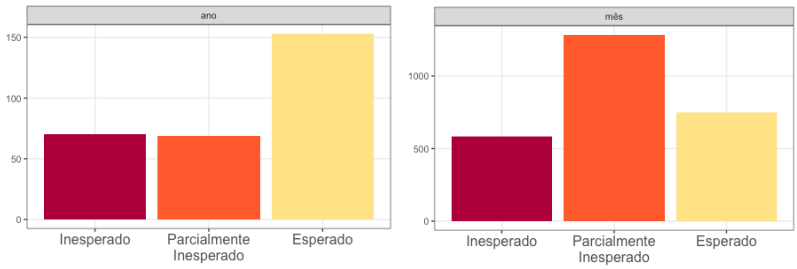


Figura 1. Número de regras por classificação sob perspectiva de mês e ano.

Para fazer uma análise das regras, elas foram ordenadas de forma crescente pelo valor de RNE e número de termos na condição, e de forma decrescente de acordo com o $lift$. Com essa ordenação, espera-se que regras inesperadas e mais generalistas estejam no topo. Seguindo esse critério, as regras classificadas como mais relevantes tem informações de datas ou de bairros. Em relação aos bairros, destacam-se Vila Militar, Jardim Sulacap, São Cristóvão, Manguinhos e Guaratiba com grande número de anomalias que indicaram velocidade inferior ao intervalo típico. Vila Valqueire, por sua vez, possui anomalias que indicaram velocidade superior ao intervalo típico, com 92% de confiança.

Por meio da análise das regras que contém informação de data na condição, a

regra classificada como mais relevante é o dia 29/06/2014 (domingo), que tem como consequência a velocidade inferior ao intervalo típico. A regra, além de ir contra o tipo de anomalia esperada para um domingo, possui *lift* $= 2,17$ e 91% de confiança. Uma análise sobre reportagens referentes a data em questão indicou que, além de estar ocorrendo a Copa do Mundo, houve um protesto na zona sul do Rio de Janeiro e uma operação conjunta da Secretaria de Ordem Pública e Secretaria de Municipal de Transportes de fiscalização de táxis na Rodoviária Novo Rio.

5. Conclusão

Este trabalho propõe uma metodologia para identificar e classificar as anomalias no comportamento do trânsito analisadas por agregações espaço-temporais. Foram usados dados da cidade do Rio de Janeiro. A metodologia proposta identificou características das principais anomalias e classificou-as como esperadas ou inesperadas. A aplicação da identificação de regras esperadas se mostrou eficiente não apenas para classificação das anomalias pontuais, como para entendimento de comportamentos sistêmicos do trânsito. Existem ainda, oportunidades para trabalhos futuros com a aplicação da metodologia proposta em outros conjuntos de dados, como dados de sistemas de transporte privado e táxis. Diferentes técnicas de identificação de anomalias também podem ser exploradas a fim entender como elas podem impactar nos resultados e nas análises.

Referências

Aggarwal, C. C. (2016). *Outlier Analysis*. Springer, New York, NY, 2nd edition.

Bierlaire, M., Chen, J., and Newman, J. (2013). A probabilistic map matching method for smartphone GPS data. *Transportation Research Part C: Emerging Technologies*, 26:78–98.

Chen, W., Guo, F., and Wang, F.-Y. (2015). A survey of traffic data visualization. *Intelligent Transportation Systems, IEEE Transactions on*, 16(6):2970–2984.

Cressie, N. and Wikle, C. K. (2015). *Statistics for spatio-temporal data*. John Wiley & Sons.

Cruz, A. B., Ferreira, J., Monteiro, B., Coutinho, R., Porto, F., and Ogasawara, E. (2017). Detecção de anomalias no transporte rodoviário urbano. In *Proceedings of the 32nd Brazilian Symposium on Databases (SBBD)*, pages 240–245.

Ferreira, N., Poco, J., Vo, H. T., Freire, J., and Silva, C. T. (2013). Visual exploration of big spatio-temporal urban data: A study of new york city taxi trips. *Visualization and Computer Graphics, IEEE Transactions on*, 19(12):2149–2158.

Giannotti, F., Nanni, M., Pinelli, F., and Pedreschi, D. (2007). Trajectory pattern mining. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 330–339.

Han, J., Kamber, M., and Pei, J. (2011). *Data Mining: Concepts and Techniques*. Morgan Kaufmann, Haryana, India; Burlington, MA, 3 edition.

Lakhina, A., Crovella, M., and Diot, C. (2004). Diagnosing network-wide traffic anomalies. In *ACM SIGCOMM Computer Communication Review*, pages 219–230. ACM.

Liu, B., Hsu, W., Chen, S., and Ma, Y. (2000). Analyzing the subjective interestingness of association rules. *IEEE Intelligent Systems and their Applications*, 15(5):47–55.

Mcgarry, K. (2005). A survey of interestingness measures for knowledge discovery. *The Knowledge Engineering Review*, 20(1):39–61.

Tao, Y., Kollios, G., Considine, J., Li, F., and Papadias, D. (2004). Spatio-temporal aggregation using sketches. In *Proceedings - International Conference on Data Engineering*, volume 20, pages 214–225.

United Nations (2014). World urbanization prospects. https://www.un-ilibrary.org/content/publication/527e5125-en.

Arquitetura para cocuradoria de dados de conhecimento popular integrados por meio de *Linked Open Data*

Marcela Mayumi Mauricio Yagui, Adriana S. Vivacqua

Universidade Federal do Rio de Janeiro – Rio de Janeiro, RJ – Brasil

marcelayagui@ufrj.br, avivacqua@dcc.ufrj.br

Abstract. *Popular knowledge databases are becoming essential for the preservation of a region's culture. In the Brazilian scenario, many museums and galleries don't incorporate systems that exploit collective intelligence in order to aid the recording of empirical information. The goal of this work is to present an architecture to support the co-curation of data derived from popular knowledge. Our proposal also provides the interconnection between visitors' contributions with data already available in open repositories on the web (in the Linked Open Data format). From this, it is hoped to achieve the enrichment of the cultural heritage of museums and their galleries, benefiting the community with access to enriched and cured knowledge databases.*

Resumo. *Bases de conhecimento popular estão se tornando essenciais para a preservação da cultura de uma região. No cenário Brasileiro, muitos museus e galerias não incorporam sistemas que exploram a inteligência coletiva de modo a auxiliar o registro de informações empíricas. Este trabalho tem como objetivo apresentar uma arquitetura para apoiar a cocuradoria de dados advindos do conhecimento popular. Nossa proposta também propicia a interconexão entre contribuições de visitantes com dados já dispostos em repositórios abertos na web (no formato Linked Open Data). A partir disso, espera-se conseguir o enriquecimento do patrimônio cultural de museus e suas galerias, beneficiando a comunidade com acesso a bases de conhecimento enriquecido e curado.*

1. Introdução

O *crowdsourcing* é um campo da *Crowd Computing* que surgiu como solução para problemas de gestão da informação por meio de geração de conteúdo por usuários na web social, onde há mobilizações e contribuições de multidões virtuais para encontrar e reunir informações. Através do *crowdsourcing* pode ser adotada a abordagem *co-curation* (em português, cocuradoria), específica para contextos aplicados ao Patrimônio Cultural (PC). A cocuradoria é derivada do termo curadoria, mas aplicada em ambientes colaborativos on-line, onde os participantes dessas atividades de cura fazem parte de uma multidão virtual [Oomen and Aroyo 2011].

Por meio do *crowdsourcing* – e consequentemente, da cocuradoria – é possível fomentar a participação do público para construir bases de conhecimento popular, que atualmente são indispensáveis para a preservação do PC de uma região. Para Cotterill *el al.* (2016) o registro do conhecimento e a digitalização de informações podem prolongar

a vida útil de objetos e histórias, tornando-as conhecidas por outras pessoas, como também preservadas para as gerações futuras.

A fim de apoiar a criação e manutenção de bases de conhecimento popular, este trabalho propõe uma arquitetura que apoia as atividades de cocuradoria deste tipo de PC, por meio da criação de chamadas *crowdsourcing* direcionadas à geração de conteúdo por visitantes. Além disso, esses dados são interligados com outros já dispostos em repositórios *Linked Open Data* (LOD), ampliando as possibilidades de recuperação, integração e publicação de conteúdo on-line. Neste trabalho buscou-se contribuir com a definição de uma arquitetura que reúne fontes de dados heterogêneas (conjunto de dados LOD e dados de contribuições de usuários) com a respectiva curadoria dessas informações.

2. Trabalhos relacionados

Estudos que utilizam a cocuradoria para assegurar o registro de informações de objetos do PC baseiam-se na mesma abordagem empregada neste trabalho. Como é o caso da plataforma Co-Curate, que integra contribuições de usuários com diversos tipos de materiais *open access* em um ambiente de museu virtual [Cotterill et al. 2016]. O trabalho de Rotman *et al.* (2012) apresentou um estudo que teve como objetivo entender as principais características de sistemas *crowdsourcing* relacionados à curadoria de conteúdo, destacando em sua análise o caso da Encyclopedia of Life. Diferente dos estudos mencionados, nosso trabalho utiliza técnicas da web semântica para interligar bases de dados e permitir a interoperabilidade entre sistemas, de modo a ampliar a recuperação de dados curados, além de permitir que o modelo de dados seja extensível.

Com relação às plataformas que utilizam *crowdsourcing* e LOD, pode-se citar o projeto Linked Jazz, que aplica a tecnologia LOD para curadoria de materiais de arquivamento digital no domínio das artes musicais. No projeto foram utilizadas técnicas de processamento de linguagem natural para reconhecimento e extração da rede de artistas da DBpedia, e o aprimoramento das conexões entre os músicos por meio de *crowdsourcing* [Pattuelli and Miller 2015]. O estudo de Pattuelli e Miller (2015) cura apenas material disponível em LOD e permite o aprimoramento desse material, não suportando a criação ou a adição de novos conteúdos. Nossa proposta utiliza o *crowdsourcing* com a abordagem *co-curation* em conjunto com a tecnologia LOD, para criar e curar diversos tipos de materiais com infraestrutura aberta.

3. Arquitetura

Esta seção apresenta a arquitetura proposta para este trabalho, que é dividida em cinco camadas: usuário, aplicação, transformação, integração e dados, explicadas a seguir. A Figura 1 ilustra a arquitetura e identifica os principais componentes envolvidos.

Camada de usuário: É composta pelos componentes ‘Contribuintes’ e ‘Curadores’. O componente ‘Contribuintes’ representa os usuários que visitam um museu e que têm a possibilidade de utilizar um dispositivo móvel para escanear um *QR Code* e recuperar informações sobre um objeto de uma ‘Coleção curada’. O componente ‘Curadores’ é formado pelos usuários especialistas que organizam de forma colaborativa assimétrica o conteúdo gerado pelos visitantes. Um usuário ‘Curador’ cria uma chamada *crowdsourcing* contendo o tema de uma coleção que fica aberta para ‘Contribuições’.

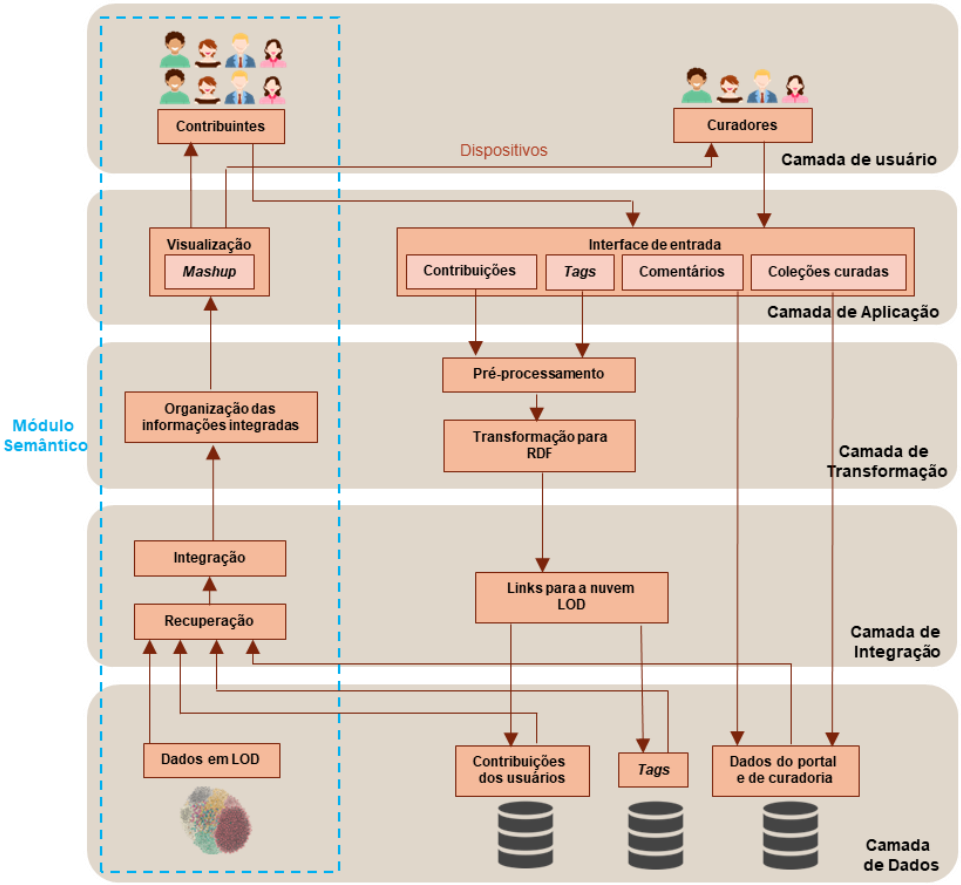


Figura 1 - Arquitetura do sistema

Camada de aplicação: É responsável pela geração do *front-end* da aplicação. Esta camada é dividida nos componentes 'Interface de entrada' e 'Visualização'. O componente 'Interface de entrada' é responsável por fornecer um meio pelo qual os usuários interagem com o sistema para a entrada de dados. Os usuários 'Contribuintes' fazem a entrada de dados das 'Contribuições', dos 'Comentários' e das '*Tags*' (componentes da arquitetura). As 'Contribuições' podem ser expressas em texto ou imagem, podendo ter tipos variados, como por exemplo, impressões pessoais, histórias, curiosidades sobre o tema da coleção ou qualquer outro tipo de material. Ou seja, por meio de *microtasking* (envio de pequenas tarefas/contribuições) o conteúdo é gerado pelo usuário com o envio de objetos para determinadas coleções.

Os 'Curadores' fazem a organização das informações com base em sua experiência sobre o tema da coleção, como indicado no componente 'Coleções curadas'. Eles analisam cada pequena 'Contribuição' para evitar que informações incorretas sejam adicionadas às coleções. Se um objeto estiver válido, ele fica disponível entre os objetos candidatos que podem ser selecionados no momento da curadoria. Caso contrário, o autor do objeto é notificado para realizar a correção de sua 'Contribuição'.

Após a validação das 'Contribuições', inicia-se o processo de curadoria das coleções. A primeira fase do processo está relacionada com a busca de conteúdo relevante para um determinado tópico escolhido para curadoria, sendo que este conteúdo é recomendado preferencialmente a partir das 'Contribuições' dos visitantes e dos dados dispostos em bases LOD. Para facilitar a busca dessas informações, são utilizadas as '*Tags*'

que foram associadas às 'Contribuições'. A partir disso, o 'Curador' seleciona, define e reorganiza as 'Coleções' usando sua experiência, ou seja, adiciona conexões entre os objetos para atribuir sentido à coleção.

Outro componente da camada de aplicação é chamado 'Visualização' (presente no Módulo Semântico). Neste componente ocorre a disponibilização das interfaces utilizando o '*Mashup'* de dados selecionados pelos 'Curadores', onde essas informações são estruturadas em *Resource Description Framework in Attributes* (RDFa) (código RDF embutido no HTML). Por meio de *QR Codes*, é possível acessar cada página e visualizar as 'Coleções'. Por meio deste componente, um 'Contribuinte' poderá reiniciar o processo de criação de conteúdo, e os 'Curadores' podem realizar novamente as tarefas de cura das 'Coleções'.

Camada de transformação: Esta camada tem como objetivo preparar os dados para posterior carga no banco de dados (componentes 'Pré-processamento' e 'Transformação para RDF') e na organização de interfaces obtidas por meio da integração dos dados (componente 'Organização das informações integradas').

O 'Pré-processamento dos dados' é o componente no qual os dados das 'Contribuições' e das '*Tags*' são analisados e limpos, sendo eliminados caracteres especiais, de modo a assegurar a qualidade das informações. Após o 'Pré-processamento', os dados são transportados para o componente 'Transformação' e convertidos em triplas RDF, com a sintaxe sujeito-predicado-objeto, onde o sujeito é uma URI, o objeto pode ser uma URI, um *blank node* ou um literal e o predicado é uma URI que define o relacionamento entre sujeito e predicado. O RDF foi utilizado pois favorece a integração de dados de diferentes esquemas, além de ser extensível, por suportar a evolução do modelo sem necessitar a alteração dos dados ou da sua estrutura.

O componente 'Organização das informações integradas' (presente no Módulo Semântico) consiste na geração de uma interface de visualização das informações curadas (provenientes das bases de dados integradas), que posteriormente são exibidas na camada de aplicação.

Camada de integração: É composta pelos componentes 'Links para a nuvem LOD', 'Recuperação' e 'Integração'. No componente 'Links para a nuvem LOD', são criadas ligações entre as triplas RDF com elementos da nuvem LOD por meio do Apache Stanbol. O Stanbol é um software *open source* cuja principal finalidade é acrescentar serviços semânticos em sistemas de gerenciamento de conteúdo, fornecendo componentes que processam conteúdo de linguagem natural em metadados RDF. Após as ligações criadas, os dados são enviados aos seus respectivos repositórios locais.

O componente 'Recuperação' (presente no Módulo Semântico) é responsável pela recuperação de dados das quatro bases que estão dispostas na camada de dados. Para a base LOD, base de 'Contribuição' dos usuários e base de '*Tags*' os dados são recuperados com a implementação de consultas federadas na linguagem SPARQL. Para a base de 'Dados do portal e de curadoria', os dados são extraídos por meio de consulta SQL no repositório relacional para posterior processamento e integração.

O componente 'Integração' (presente no Módulo Semântico) tem como objetivo integrar os 'Dados do portal e de curadoria', dispostos no formato relacional, com os dados em RDF e em LOD obtidos por meio de consultas SPARQL.

Camada de dados: É composta por três bases de dados armazenadas localmente e uma base de dados externa. A primeira base contém 'Dados do portal, administrativas do sistema e de curadoria' e 'Comentários', e é armazenada no formato relacional no SGBD MySQL. A segunda base corresponde aos dados das 'Contribuições' dos usuários, dos objetos textuais e imagens. A terceira base contém as '*Tags*' que descrevem as 'Contribuições', a fim de enriquecer as descrições e melhorar a recuperação dos dados. A segunda e terceira base são armazenadas em RDF no banco de triplas Apache Jena Fuseki. Por fim, a base externa é composta pela nuvem LOD, disponível na web, onde informações relacionadas aos objetos de contribuições são selecionadas posteriormente.

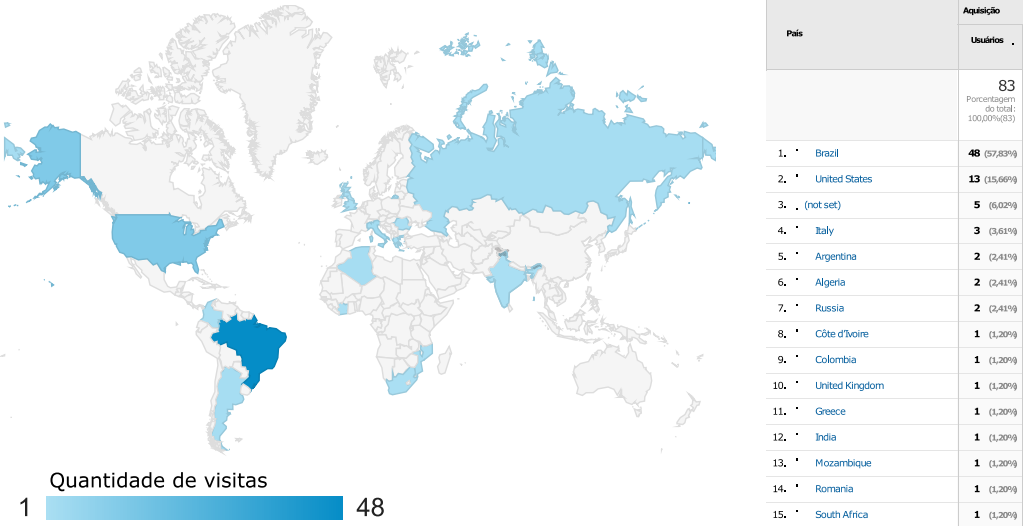
O Módulo Semântico [Yagui et al. 2017b] (destacado em azul) está indicado na arquitetura por representar o caminho pelo qual os dados em LOD percorrem até a formação de um conteúdo curado estar disponível para ser consumido por um usuário.

4. Prova de conceito

Para a aplicação prática do Módulo Semântico, foram utilizadas bases de dados abertas, disponíveis na web, relacionadas a plantas medicinais e instituições curadoras relacionadas à Botânica. Nosso objetivo foi testar o Módulo Semântico para realizar a curadoria de conteúdo aberto e disponibilizá-la por meio de um aplicativo web, de modo a permitir que visitantes possam consumir as informações curadas.

O aplicativo recupera e integra dados de variadas fontes, entre elas: a DBpedia, o Bio2RDF e o Global Biodiversity Information Facility (GBIF). O GBIF possui um acervo similar à proposta apresentada deste trabalho, de modo que ao realizar testes no Módulo Semântico com seus dados, esperou-se que o resultado pudesse ser o mais fidedigno possível a uma aplicação real.

O aplicativo foi testado de modo presencial em duas ocasiões diferentes, entre os meses de outubro [Yagui et al. 2017a] e novembro [Yagui et al. 2017b] de 2017. Em ambas as ocasiões, o aplicativo foi apresentado a visitantes em exposições do tipo 'feira de ciências'. Para avaliar a experiência dos visitantes, nós os convidamos a expressar sua opinião durante e após a utilização. Em geral, o aplicativo foi bem aceito: em particular, os usuários apreciaram a facilidade em acessar informações das plantas a partir do



País	Aquisição Usuários
	83 Porcentagem do total: 100,00% (83)
1. Brazil	48 (57,83%)
2. United States	13 (15,66%)
3. (not set)	5 (6,02%)
4. Italy	3 (3,61%)
5. Argentina	2 (2,41%)
6. Algeria	2 (2,41%)
7. Russia	2 (2,41%)
8. Côte d'Ivoire	1 (1,20%)
9. Colombia	1 (1,20%)
10. United Kingdom	1 (1,20%)
11. Greece	1 (1,20%)
12. India	1 (1,20%)
13. Mozambique	1 (1,20%)
14. Romania	1 (1,20%)
15. South Africa	1 (1,20%)

Figura 2 – Frequência de utilização do aplicativo

leitor de *QR Code*. Em contraste, foram relatadas algumas dificuldades para acessar informações de espécimes e institutos curadores no mapa. Adicionalmente, ao final das exibições disponibilizamos uma página que contém todos os *QR Codes* com links para as plantas e convidamos os usuários a compartilhar o aplicativo em suas redes sociais. Deste modo, foi possível obter dados sobre a utilização do aplicativo de modo não presencial, através de JavaScripts do Google Analytics adicionalmente implementados. Embora nesta última modalidade não tenham sido realizadas experimentações acerca da satisfação dos usuários, foi possível constatar que houve interesse pelo *App*, tendo o mesmo sido utilizado por diversos usuários, conforme mostra a Figura 2.

Detalhes adicionais sobre o Módulo Semântico estão relatados em [Yagui et al. 2017b].

5. Conclusão e Trabalhos Futuros

Este trabalho apresentou uma arquitetura para dar suporte à criação e à manutenção de bases de conhecimento popular. Com a finalidade de apoiar as atividades de cocuradoria deste tipo de PC, a arquitetura faz uso de dois mecanismos de interação: (i) criação de chamadas *crowdsourcing* direcionadas à criação de conteúdo por visitantes e (ii) cura desses materiais por especialistas. A arquitetura também permite que os dados sejam interligados com outros já dispostos em repositórios LOD, ampliando a possibilidade de recuperação de conteúdo relacionado.

Além disso, foi realizada uma prova de conceito do Módulo Semântico da arquitetura com aplicação de bases de dados abertas criadas colaborativamente no domínio da Botânica. Desta forma, a proposta deste módulo mostrou-se adequada para o cenário de aplicação no qual este trabalho foi projetado, de modo que há suporte para a inclusão futura de outros dados empíricos gerados por visitantes. Como trabalhos futuros, serão realizados estudos baseados em testes de usabilidade e avaliações qualitativas com curadores de instituições culturais.

Referências

Cotterill, S., Hudson, M., Lloyd, K., et al. (2016). Co-curate: Working with Schools and Communities to Add Value to Open Collections. *Journal of Interactive Media in Education*, v. 2016, n. 1.

Oomen, J. and Aroyo, L. (2011). Crowdsourcing in the cultural heritage domain: opportunities and challenges. In *Proceedings of the 5th International Conference on Communities and Technologies*.

Pattuelli, M. C. and Miller, M. (2015). Semantic network edges: a human-machine approach to represent typed relations in social networks. *Journal of Knowledge Management*, v. 19, n. 1, p. 71–81.

Rotman, D., Procita, K., Hansen, D., Parr, C. S. and Preece, J. (2012). Supporting content curation communities: The case of the Encyclopedia of Life. *Journal of the American Society for Information Science and Technology*, v. 63, n. 6, p. 1092–1107.

Yagui, M. M. M., Maia, L. F. M. P., Oliveira, J. and Vivacqua, A. S. (2017a). Applying Linked Open Data and ETL for Mapping and Visualization of Physical Objects in Botany. In *Anais do XXIII Simpósio Brasileiro de Sistemas Multimídia e Web: Workshops e Pôsteres*.

Yagui, M. M. M., Maia, L. F. M. P., Oliveira, J. and Vivacqua, A. S. (2017b). Curation of Physical Objects in Botany: Architecture and Development of a Linked Open Data-Based Application. In *Proceedings of the 2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)*. IEEE.

Utilização de Redes Heterogêneas para Medir a Força dos Relacionamentos no GitHub

Gabriel P. Oliveira, Natércia A. Batista, Michele A. Brandão, Mirella M. Moro

Universidade Federal de Minas Gerais (UFMG) – Belo Horizonte, MG – Brasil

{gabrielpoliveira,natercia,micheleabrandao,mirella}@dcc.ufmg.br

Abstract. *Our goal is to measure the strength of the relationships between GitHub users by considering social and technical features. The contributions include a new heterogeneous graph model with different types of interactions and new metrics for the strength of such relationships. The results show the proposed metrics bring new information about the relationships.*

Resumo. *Nosso objetivo é medir a força dos relacionamentos entre usuários do GitHub considerando fatores técnicos e sociais. As contribuições incluem uma nova modelagem de grafo heterogêneo com diferentes tipos de interação e novas métricas para a força de tais relacionamentos. Os resultados mostram que as métricas propostas acrescentam informação sobre os relacionamentos.*

1. Introdução

É tarefa primordial da área de Bancos de Dados agregar ou enriquecer dados existentes a fim de extrair informações relevantes e até novos conhecimentos a partir dos mesmos. De fato, essa área (como outras da Computação) tem evoluído para abranger as mais diversas necessidades de processamento de dados atuais. Por exemplo, com a expansão da Internet e o acesso a serviços de banda larga, pessoas comuns têm à disposição uma gama de aplicativos e serviços online. Com tal acesso, geram-se diariamente grandes volumes de dados que podem ser processados com os mais variados propósitos. Entre tantas opções, um dos serviços mais populares é o de redes sociais online, que conectam pessoas a partir de seus relacionamentos pessoais e profissionais.

Neste trabalho, estudamos o GitHub, uma rede de desenvolvimento colaborativo de software. Especificamente, uma modelagem de rede heterogênea para representar diferentes tipos de colaboração e novas formas de medir a força dos relacionamentos são propostas, considerando fatores técnicos e sociais. Assim, este trabalho propõe métricas semânticas que consideram tais fatores e analisa se elas confirmam a força dos relacionamentos definida por métricas existentes ou adicionam novas informações. Em termos de aplicação, tais métricas permitem descoberta de padrões que auxiliam no estudo da formação de times, detecção de comunidades e identificação de influentes, entre outras.

2. Trabalhos Relacionados

Em redes sociais, existem diversos estudos sobre força de relacionamentos [Alves et al. 2016; Casalnuovo et al. 2015; Goyal et al. 2018]. Exemplos de métricas específicas estão nas Tabelas 1 e 2. Porém, nenhum diferencia a força dos relacionamentos em fatores técnicos e sociais em rede *heterogênea* a partir do GitHub. Tais estudos consideram apenas um fator (e.g., a intensidade de contribuições em um repositório) para determinar tal

Tabela 1. Métricas definidas por meio de propriedades topológicas.
Considere para um nó X da rede, $\mathcal{N}(X)$ como o conjunto de vizinhos de X, $w(X)$ como a soma dos pesos das arestas conectadas a X e $w(X,Y)$ como o peso da aresta entre X e Y.

Métricas topológicas	
Neighborhood Overlap (NO) - [Easley and Kleinberg 2010]	É uma maneira de medir a força das ligações entre nós por meio da similaridade de seus vizinhos: $NO_{(X,Y)} = \frac{\lvert\mathcal{N}(X)\cap\mathcal{N}(Y)\rvert}{\lvert\mathcal{N}(X)\cup\mathcal{N}(Y)-\{X,Y\}\rvert}$.
Preferencial Attachment (PA) - [Barabási and Albert 1999]	Há uma relação linear entre o número de vizinhos de um nó e a sua probabilidade de conectar-se a outro nó: $PA_{(X,Y)} = \lvert\mathcal{N}(X)\rvert\lvert\mathcal{N}(Y)\rvert$.
Adamic-Adar [2003] (AA)	Dá maior peso aos vizinhos que não se relacionam com muitos outros: $AA_{(X,Y)} = \sum_{\forall Z\in\mathcal{N}(X)\cap\mathcal{N}(Y)} \frac{1}{log\lvert\mathcal{N}(Z)\rvert}$.
Métricas topológicas ponderadas	
Tieness (T)* - [Brandão and Moro 2017]	Mede a força das relações de coautoria; no contexto do GitHub: $T_{(X,Y)} = \frac{\frac{\lvert\mathcal{N}(X)\cap\mathcal{N}(Y)\rvert+1}{1+\lvert\mathcal{N}(X)\cup\mathcal{N}(Y)-\{X,Y\}\rvert}\lvert\lvert w(X,Y)\rvert\rvert}{2}$, onde o valor do peso $w(X,Y)$ é normalizado.

** Utilizada em conjunto com cada uma das métricas semânticas da Tabela 2, cujo valor é o peso $w(X,Y)$.*

Tabela 2. Métricas definidas por meio de propriedades semânticas.
Propostas por Alves et al. [2016] e Batista et al. [2017a], consideram a semântica das relações no contexto específico do GitHub. Seja $\mathcal{R}$ o conjunto de todos os repositórios onde dois usuários X e Y colaboram.

Métricas semânticas no contexto do GitHub	
Number of Shared Repositories (SR)	Definida como o número de repositórios compartilhados entre dois desenvolvedores: $SR_{(X,Y)} = \lvert\mathcal{R}\rvert$.
Jointly Developers Contribution to Shared Repositories (JCSR)	Seja $JCSR_{(X,Y,r_i)}$ a razão entre a quantidade de desenvolvedores no par e o total de desenvolvedores em r_i, $JCSR_{(X,Y)} = \frac{\sum_{\forall r_i\in\mathcal{R}} JCSR_{(X,Y,r_i)}}{\lvert\mathcal{R}\rvert}$.
Jointly Developers Commits to Shared Repositories (JCOSR)	Sejam $NC_{(X,r_i)}$ o número de *commits* feitos por um usuário X em um repositório r_i e $NC_{(r_i)}$ o número total de *commits* no repositório r_i: $JCOSR_{(X,Y)} = \sum_{\forall r_i\in\mathcal{R}} \frac{NC_{(X,r_i)}+NC_{(Y,r_i)}}{NC_{(r_i)}}$.
Previous Collaboration (PC)	Seja $ND_{(r_i,t)}$ o número de desenvolvedores no repositório r_i no tempo t, então a quantidade de colaboração de X e Y em t: $PC_{(X,Y,t)} = \frac{\sum_{\forall r_i\in\mathcal{R}} \frac{1}{ND_{(r_i,t)}}}{\lvert\mathcal{R}\rvert}$.
Global Potential Contribution (GPC)	Seja $T_{(X,Y,r_i)}$ o intervalo de tempo em que os desenvolvedores X e Y contribuem no repositório r_i e $\mathcal{D}$ o conjunto de todos os desenvolvedores na rede: $GPC_{(X,Y)} = \frac{\sum_{\forall r_i\in\mathcal{R}} T_{(X,Y,r_i)}}{max_{\forall(D_i,D_j)\in\mathcal{D},r_i\in\mathcal{R}} T_{(D_i,D_j,r_i)}}$.

força em uma rede homogênea. Assim, a modelagem *heterogênea* proposta aqui permite uma definição mais ampla e completa da força dos relacionamentos.

3. Rede Social de Colaboração

Esta seção descreve a base de dados utilizada para criar a rede social e a modelagem da rede heterogênea de desenvolvimento colaborativo de software.

Base de Dados. A base de dados utilizada é originada do GitSED 2015 (*GitHub Socially Enhanced Dataset*) [Batista et al. 2017b], um conjunto de dados do GitHub curado, expandido e enriquecido a partir do GHTorrent [Gousios 2013]. A versão original do GitSED considera repositórios desenvolvidos em apenas três linguagens de programação. Dessa forma, expandimos a base de dados para considerar seis linguagens subdivididas

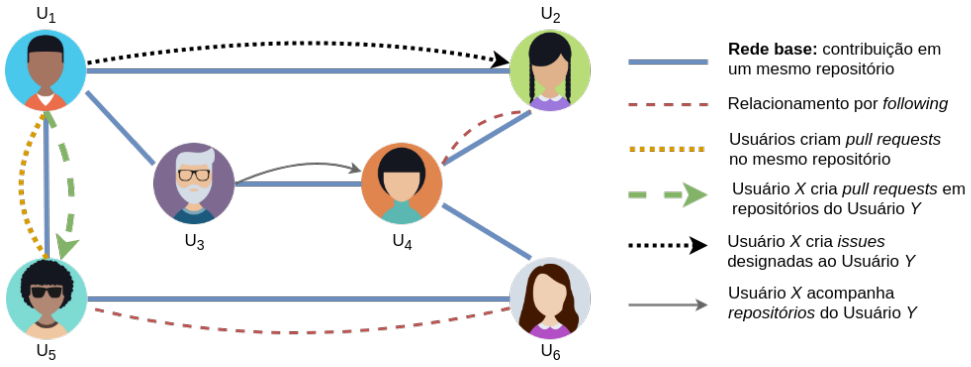


Figura 1. Exemplo da rede heterogênea de colaboração do GitHub com novos tipos de relacionamento considerados.

em dois grupos, de acordo com seu nível de colaboração definido por Rocha et al. [2016]: linguagens mais colaborativas (JavaScript, Ruby e Python) e linguagens menos colaborativas (Assembly, Pascal e Visual Basic). Utilizando a mesma metodologia de coleta e curadoria, atualizou-se o GitSED com dados do GHTorrent até maio de 2017. Portanto, tem-se uma versão *ampliada* do conjunto de dados, disponibilizada publicamente[1].

Modelagem da Rede. A partir da rede base com relacionamentos de colaboração no mesmo repositório, a nova modelagem contém seis (dos quais cinco são novos e aqui propostos) tipos de arestas: (i) colaboração no mesmo repositório (obrigatório, pois são desenvolvedores-colaboradores); (ii) seguidores; (iii) criação de *pull requests* em um mesmo repositório; (iv) criação de *pull requests* em repositório de outro usuário; (v) criação de *issues* designadas a outro usuário; e (vi) acompanhamento de repositórios favoritos. Assim, a rede se torna *heterogênea* representada por um multigrafo $\mathcal{G}' = (\mathcal{V}, \mathcal{E}_1 \cup \mathcal{E}_2 \cup \cdots \cup \mathcal{E}_6)$, onde o conjunto $\mathcal{V}$ continua com vértices para os desenvolvedores da rede; porém, cada conjunto $\mathcal{E}_k$, para $k \in \{1, \cdots, 6\}$, representa um dos seis tipos de arestas. Ademais, o peso das arestas também é calculado por métricas topológicas descritas na Tabela 1, ou semânticas na Tabela 2.

4. Novas Métricas Semânticas

A Figura 1 ilustra os seis tipos de relacionamentos presentes na nova modelagem heterogênea do GitHub. Então, a Tabela 3 apresenta as métricas propostas a partir de propriedades semânticas no contexto do GitHub para medir a força dos relacionamentos.

5. Caracterização das Redes, Análise e Correlação das Métricas

Esta seção apresenta os resultados obtidos na caracterização das redes de colaboração de cada linguagem e no estudo da correlação das diferentes métricas.

Caracterização das Redes Heterogêneas de Colaboração. As seis linguagens de programação foram analisadas para verificar se o nível de colaboração de cada uma se mantém em relação à classificação feita por Rocha et al. [2016] (note que o dataset foi *expandido*). As Tabelas 4 e 5 mostram os resultados dessa caracterização. Para as mais colaborativas (Tabela 4), as três possuem um alto índice de nós e arestas em seus maiores componentes conectados (*Giant Component* – GC). Em todas elas, mais de 80% das

[1]Projeto Apoena: `http://www.dcc.ufmg.br/~mirella/projs/apoena`

Tabela 3. Novas métricas para a força dos relacionamentos no GitHub.
Fatores técnicos conectam desenvolvedores por meio dos aspectos de desenvolvimento de software, enquanto que fatores sociais consideram aspectos da interação *direta* entre dois usuários no GitHub.

Novas métricas semânticas considerando fatores técnicos	
Unidirectional Assigned Issues (UAI)	Sejam $NI_{(X,Y)}$ o número de *issues*[2] criadas pelo usuário X que são designadas ao usuário Y e $NTI_{(Y)}$ o número total de *issues* designadas ao usuário Y: $UAI_{(X,Y)} = \frac{NI_{(X,Y)}}{NTI_{(Y)}}$.
Unidirectional Pull Requests (UPR)	Sejam $PR_{(X,Y)}$ o número de *pull requests*[3] que o usuário X cria em repositórios do usuário Y e $TPR_{(Y)}$ o número total de *pull requests* que os repositórios do usuário Y possuem: $UPR_{(X,Y)} = \frac{PR_{(X,Y)}}{TPR_{(Y)}}$.
Bidirectional Pull Requests (BPR)	Sejam $NPR_{(X,r)}$ o número de *pull requests* que o usuário X cria em um repositório r, $NTPR_{(r)}$ o número total de *pull requests* em r e $\mathcal{U}$ o conjunto universo dos repositórios existentes na base. Para cada par (X, Y) em um repositório $r \in \mathcal{U}$, se $NPR_{(X,r)} \neq 0$ e $NPR_{(Y,r)} \neq 0$: $BPR_{(X,Y)} = \sum_{\forall r_i \in \mathcal{U}} \frac{NPR_{(X,r_i)} + NPR_{(Y,r_i)}}{NTPR_{(r_i)}}$.
Novas métricas semânticas considerando fatores sociais	
Bidirectional Intensity of Followers (BIF)	A intensidade de seguidores é definida com base na relação onde um usuário X *segue* um usuário Y no GitHub. Propõe-se os seguintes valores para medir tal intensidade: $BIF_{(X,Y)} = \begin{cases} 1 & \text{se } X \text{ segue } Y \textbf{ AND } Y \text{ segue } X \\ 0,5 & \text{se } X \text{ segue } Y \textbf{ XOR } Y \text{ segue } X \\ 0 & \text{caso contrário} \end{cases}$
Unidirectional Intensity of starMarks (UIM)	Sejam $NS_{(X,Y)}$ o número de repositórios de um usuário Y nos quais X tem interesse (clicando no botão *star*) e $NRS_{(X)}$ o número total de repositórios nos quais X está interessado: $UIM_{(X,Y)} = \frac{NS_{(X,Y)}}{NRS_{(X)}}$.

arestas da rede estão presentes no GC. Então tais redes são bem conectadas, confirmando a classificação de Rocha et al. [2016]. Para menos colaborativas (Tabela 5), as três possuem comportamentos semelhantes em suas redes, todas com baixo grau médio e baixo índice de arestas no GC. Portanto, infere-se que os repositórios dessas linguagens são em sua maioria compostos por poucos desenvolvedores, que colaboram pouco entre si. Devido a restrições de espaço, as análises seguintes consideram Ruby e Visual Basic como representantes das linguagens mais e menos colaborativas, respectivamente.

Análise das Novas Métricas Semânticas. Para verificar a independência entre as novas métricas, a correlação entre elas é analisada por meio dos coeficientes de Pearson e Spearman (verificam relações lineares e monotônicas entre as métricas, respectivamente). Por limitações de espaço, os resultados das correlações são apenas discutidos. Em todas as linguagens, observa-se que a correlação entre as novas métricas é baixa ou insubstancial, com valores próximos a zero. Tal resultado pode ser explicado pelo fato de que cada métrica considera fatores diferentes para calcular a interação entre os pares. A exceção está nas métricas UPR e BPR, que analisam a interação por meio da criação de *pull requests*. A baixa correlação entre as métricas é um forte indicador de que as propriedades semânticas consideradas adicionam novas informações à rede de colaboração do GitHub.

Para entender o quanto cada tipo de relacionamento representa da rede, a Tabela 6 mostra a proporção de desenvolvedores e arestas presentes quando são retirados os pares que não contêm o tipo de relacionamento representado por cada métrica. Os resultados mostram que poucos nós e arestas permanecem na rede para métricas com fatores sociais (BIF e UIM). Ou seja, desenvolvedores do GitHub não consideram tais funcionalidades

Tabela 4. Estatísticas das redes das linguagens mais colaborativas.

	JavaScript	Python	Ruby
Número de repositórios	6.767.297	3.074.827	2.536.133
Número de nós (desenvolvedores)	854.255	519.771	279.281
Número de arestas	2.571.154	3.699.096	33.979.590
Densidade (10^{-3})	0,007	0,027	0,871
Grau médio	6,02	14,23	243,34
Coeficiente de Clusterização Médio	0,358	0,384	0,429
Número de nós no GC*	379.637 (44,4%)	259.355 (49,9%)	180.175 (64,5%)
Número de arestas no GC*	2.105.747 (81,9%)	3.282.140 (88,7%)	33.873.748 (99,7%)

** Giant Component (GC): maior componente conectado de um grafo*

Tabela 5. Estatísticas das redes das linguagens menos colaborativas.

	Assembly	Pascal	Visual Basic
Número de repositórios	35.073	20.330	33.275
Número de nós (desenvolvedores)	7.516	3.520	5.602
Número de arestas	14.906	9.377	7.205
Densidade (10^{-3})	0,528	1,514	0,459
Grau médio	3,97	5,3	2,57
Coeficiente de Clusterização Médio	0,354	0,374	0,311
Número de nós no GC*	483 (6,4%)	577 (16,4%)	95 (1,7%)
Número de arestas no GC*	3.335 (22,3%)	5.140 (54,8%)	1.368 (19%)

** Giant Component (GC): maior componente conectado de um grafo*

Tabela 6. Participação na rede a partir de métricas selecionadas.

Métrica	# de desenvolvedores		# de arestas	
	Ruby	Visual Basic	Ruby	Visual Basic
Rede completa	279.281 (100%)	5.602 (100%)	33.979.590 (100%)	7.205 (100%)
UAI - *Unidirecional Assigned Issues*	9.982 (3,57%)	124 (2,21%)	8.517 (0,02%)	82 (1,14%)
UPR - *Unidirecional Pull Requests*	14.811 (5,3%)	166 (2,96%)	10.927 (0,03%)	104 (1,44%)
BPR - *Bidirectional Pull Requests*	57.604 (20,63%)	453 (8,09%)	1.697.256 (5%)	391 (5,43%)
BIF - *Bidirecional Intensity of Followers*	58.715 (21,02%)	923 (16,48%)	102.736 (0,3%)	616 (8,55%)
UIM - *Unidirecional Intensity of starMarks*	13.248 (4,74%)	107 (1,91%)	15.577 (0,04%)	56 (0,78%)

quando colaboram em um repositório. Em relação às métricas com fatores técnicos (UAI, UPR e BPR), o baixo número de nós e arestas é explicado pelo fato de que grande parte das *issues* e *pull requests* nos repositórios provém de usuários externos.

Correlação com as Métricas Existentes. Para analisar a correlação entre as métricas existentes (NO, PA, AA, T, PC, GPC, SR, JCSR, JCOSR) e as propostas (BPR, UAI, UPR, BIF e UIM), utilizamos o coeficiente de Pearson e Spearman. Os resultados são similares para ambos em todas as linguagens, com algumas exceções (novamente, devido a limitações de espaço, os resultados são discutidos sem gráficos ou tabelas). Não há correlação significativa linear ou monotônica entre a maioria das propriedades. Uma das exceções são AA e PA com correlação acima de 0,87 considerando todas as linguagens. Se PA e AA estão correlacionadas, então é esperado que uma métrica correlacionada com AA também esteja com PA e vice-versa. Outra exceção é JCSR (contribuição conjunta em repositórios) em correlação negativa no intervalo [-0,95;-0,57] com PA e AA para todas as linguagens. Isso pode indicar que pares de desenvolvedores com intensa contribuição conjunta em repositórios podem não se conectar com muitos outros desenvolvedores.

Uma diferença significativa entre as linguagens mais e menos colaborativas é a existência de correlação negativa no intervalo [-0,9;-0,4] entre JCOSR com GPC, PA, AA e NO para Assembly, Pascal e Visual Basic. Tal resultado indica que nas linguagens menos colaborativas, os relacionamentos de um par de desenvolvedores com seus vizinhos influenciam negativamente à contribuição conjunta por *commits*. Por outro lado, a

correlação negativa entre JCOSR e GPC indica que pares de desenvolvedores tendem a não contribuir por muito tempo. Existe também forte correlação entre as métricas *Tieness* ponderadas com propriedades semânticas, ou seja, *Tieness* com diferentes pesos traz as mesmas informações à análise dos relacionamentos, e assim, pode-se escolher apenas uma delas. Nesse caso, recomenda-se o uso de métricas com baixo custo computacional, como T_BIF ou T_SR (combinação de T com BIF e T com SR, respectivamente).

6. Conclusão e Trabalhos Futuros

Este artigo apresentou uma nova modelagem heterogênea para a rede de colaboração entre desenvolvedores no GitHub, bem como novas métricas semânticas para a força dos relacionamentos. A análise revelou que todas representam informações novas sobre os relacionamentos. Ademais, altas correlações negativas entre métricas que consideram a contribuição em repositórios e métricas baseadas em vizinhos mostraram que a colaboração é mais intensa entre pares de desenvolvedores com menos vizinhos. Como trabalho futuro, pretende-se investigar a relação dessas propriedades semânticas com o ranqueamento e a influência de desenvolvedores no GitHub. Também planeja-se investigar como fatores técnicos são influenciados por fatores sociais.

Agradecimentos. Trabalho parcialmente financiado por CAPES, CNPq e FAPEMIG.

Referências

Adamic, L. A. and Adar, E. (2003). Friends and neighbors on the Web. *Social Networks*, 25(3):211 – 230.

Alves, G. B., Brandão, M. A., Santana, D. M., da Silva, A. P. C., and Moro, M. M. (2016). The Strength of Social Coding Collaboration on GitHub. In *SBBD - Short Papers*.

Barabási, A.-L. and Albert, R. (1999). Emergence of scaling in random networks. *Science*, 286(5439):509–512.

Batista, N. A., Brandão, M. A., Alves, G. B., da Silva, A. P. C., and Moro, M. M. (2017a). Collaboration Strength Metrics and Analyses on GitHub. In *WI*, pages 170–178.

Batista, N. A. et al. (2017b). GitSED: Um Conjunto de Dados com Informações Sociais Baseado no GitHub. In *SBBD - Dataset Showcase Workshop*, pages 224–233.

Brandão, M. A. and Moro, M. M. (2017). The strength of co-authorship ties through different topological properties. *JBCS*, 23(1):5.

Casalnuovo, C. et al. (2015). Developer onboarding in GitHub: the role of prior social links and language experience. In *ESEC/FSE*, pages 817–828.

Easley, D. and Kleinberg, J. (2010). *Networks, crowds, and markets: Reasoning about a highly connected world*. Cambridge University Press.

Gousios, G. (2013). The GHTorrent Dataset and Tool Suite. In *MSR*, pages 233–236.

Goyal, R. et al. (2018). Identifying unusual commits on GitHub. *Journal of Software: Evolution and Process*, 30(1).

Rocha, L. M. A., Silva, T. H. P., and Moro, M. M. (2016). Análise da Contribuição para Código entre Repositórios do GitHub. In *SBBD - Short Papers*, pages 103–108.

A Distributed System for SearchOnMath Based on the Microsoft BizSpark Program

Ricardo M. Oliveira[1], **Flavio B. Gonzaga**[1], **Valmir C. Barbosa**[2], **Geraldo B. Xexéo**[2]

[1]DCC, UNIFAL-MG, Rua Gabriel Monteiro da Silva, 700, 37130-001 Alfenas - MG

[2]PESC, UFRJ, Caixa Postal 68511, 21941-972 Rio de Janeiro - RJ

fbgonzaga@bcc.unifal-mg.edu.br

Abstract. *Mathematical information retrieval is a relatively new area, so the first search tools capable of retrieving mathematical formulas began to appear only a few years ago. The proposals made public so far mostly implement searches on internal university databases, small sets of scientific papers, or Wikipedia in English. As such, only modest computing power is required. In this context, SearchOnMath has emerged as a pioneering tool in that it indexes several different databases and is compatible with several mathematical representation languages. Given the significantly greater number of formulas it handles, a distributed system becomes necessary to support it. The present study is based on the Microsoft BizSpark program and has aimed, for 38 different distributed-system scenarios, to pinpoint the one affording the best response times when searching the SearchOnMath databases for a collection of 120 formulas.*

Introduction

Unlike textual information retrieval, for which there exist several techniques already widely studied and disseminated, as well as tools capable of tackling the required tasks while performing quite satisfactorily, the area of Mathematical Information Retrieval (MIR) is still in a much less developed stage. In fact, as summarized in Table 1, only in the past few years have techniques for MIR been introduced, usually focusing on very specific problems related to Wikipedia's mathematical pages and indexing around 500 000 formulas.

As with most niche-oriented forms of information retrieval, MIR has to contend with problems that are specific to the search for mathematical formulas. One of them is the large overhead caused by the various possible uses for the same symbol [Schubotz et al. 2016]. These possibilities constitute an important source of ambiguity in MIR, since completely different formulas can be written using essentially the same symbols [Kamali and Tompa 2013]. Another problem is the fact that usually the formulas available on the Web are represented in languages originally conceived with little or no concern for a formula's semantic aspects.

The search engine SearchOnMath (searchonmath.com) is one of the most recent arrivals to the field of MIR. Its first version was released in 2013, and by the end of 2015 it had become a start-up. It soon joined the Microsoft BizSpark program, with a modest but very effective monthly allowance, distributed among five email accounts, to hire computers (at most 20 processing cores). Until 2016, SearchOnMath was able

Table 1. Existing Tools for MIR

Reference	Search Domain	No. of Formulas
[Kohlhase and Sucan 2006]	CONNEXIONS project; functions.wolfram.com	77 000; 87 000
[Asperti et al. 2006]	Coq proof assistant	40 000 theorems
[Pavan Kumar et al. 2012]	Database created by authors	829
[Schellenberg et al. 2012]	50 LaTeX documents	24 479
[Kamali and Tompa 2013]	en.wikipedia.org; dlmf.nist.gov	611 210; 252 148
[Hu et al. 2013]	en.wikipedia.org	495 958
[Lin et al. 2014]	en.wikipedia.org; citeseerx.ist.psu.edu	521 782; 9 482
[Stalnaker and Zanibbi 2015]	en.wikipedia.org[1]	482 364
[Zanibbi et al. 2016]	en.wikipedia.org[2]	387 947

to perform the search for formulas on four databases, namely, the English version of Wikipedia, Wolfram MathWorld, DLMF, and PlanetMath. In such a scenario, only one computer with the so-called A3 Basic configuration of the Microsoft Azure environment was sufficient. This configuration includes a four-core processor, 7 GiB of RAM, and a 120-GB HD.

In the course of 2016, as SearchOnMath began preparations for expansion, a distributed system was developed and tested on Microsoft Azure with the goal of assessing each of the 38 possible configurations afforded by our constraints within BizSpark. This investigation was based on a set of 120 preselected formulas that were to be worked on by SearchOnMath within a domain of almost 2 million formulas, and aimed at discovering which of the candidate configurations was capable of delivering the best response times. Our results and conclusions are presented in this paper.

Our study contributes to the field of MIR in two different ways. The first of them is more of a confirmation of the path we have selected for SearchOnMath. It is therefore of an immediate nature, with short-term applicability by other entrepreneurs. As companies that develop search engines for mathematical formulas begin to appear, mainly as start-ups, it may be reassuring to know that the Microsoft BizSpark program is a very viable opportunity, since it already supports more than 100 000 start-ups worldwide and continues to expand. In this regard, information about the infrastructure and operation of SearchOnMath on Azure can be more widely useful. The second contribution is the performance assessment we carried out itself, including the set of 120 formulas that we put together in order to measure response time, but which can be used for other purposes as well.

Methodology

For the present study we considered five databases, all obtained throughout the year 2016. Each of these databases is identified in Table 2, along with the respective number of mathematical formulas extracted from it, disregarding repetitions.

[1]Used MREC (Math REtrieval Collection), a collection with approximately 324 000 academic publications, during the development phase.

[2]Includes some information about index size and response time when applied to an arXiv base with about 60 million formulas.

Table 2. Database List

Database	Number of Formulas
Wikipedia, English version, en.wikipedia.org	590 417
Wolfram MathWorld, mathworld.wolfram.com	79 677
DLMF, dlmf.nist.gov	33 219
PlanetMath, planetmath.org	159 944
Socratic, socratic.org	1 063 754

After the bases were obtained individually, a final database was constructed as the union of all five, still disregarding repetitions. The resulting database contains 1 905 358 indexed formulas. SearchOnMath was then configured as in Figure 1. For operation, a client submits a formula to be searched to the master machine, which runs the engine's front-end. After reception by the master, the formula is sent to the slave machines, which do all the necessary processing to find out which formulas in the database are similar to the query formula. The database is distributed across the slaves so that, for example, if we have 10 of them, then each one has approximately 10% of the formulas. After processing, each slave returns a list containing the most similar formulas it found. The master receives all the lists and then performs the final ordering of the results, returning the consolidated list to the client.

All machines run Linux, and in all cases we configured the master machine with four cores, 7 GiB of RAM, and a 120-GB HD (this configuration is called A3 Basic in Azure). The number of slave machines was obtained based on the remaining allowance resources. We first estimated the amount of our monthly allowance that would correspond to an hour, and then took into account the fact that Azure prices the allocation of machines differently, depending on geographic location. We always chose the region that offered the lowest possible cost in the United States, considering as reference value the one quoted at the date of the beginning of the experiments (Nov. 23, 2016). In these circumstances, discounting the A3 Basic cost per hour allowed us the allocation of up to 16 cores to work as slave machines, arranged according to 38 (out of 65) different configurations available in the Azure environment.

Table 3 shows all the configurations analyzed for the slave machines. The Config. column indicates the name of the configuration, its resources detailed in the Cores, RAM

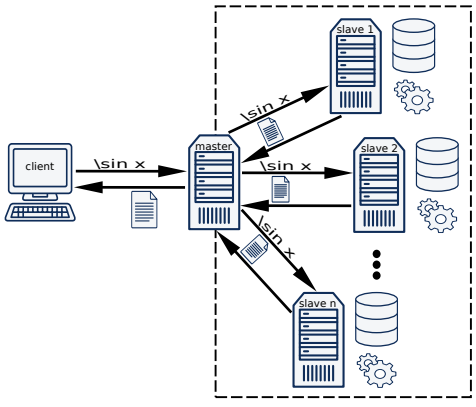


Figure 1. The SearchOnMath architecture.

Table 3. Slave-Machine Configurations and Azure Groups

Config.	Cores	RAM	HD	Mach.	Config.	Cores	RAM	HD	Mach.
A0 Basic	1	0.75	20	16	A0 Std.	1	0.75	20	16
A1 Basic	1	1.75	40	16	A1 Std.	1	1.75	70	16
A2 Basic	2	3.50	60	8	A2 Std.	2	3.50	135	8
A3 Basic	4	7.00	120	4	A3 Std.	4	7.00	285	4
A4 Basic	8	14.00	240	2	A4 Std.	8	14.00	605	2
A1 v2	1	2.00	10	16	A5 Std.	2	14.00	135	3
A2 v2	2	4.00	20	8	A6 Std.	4	28.00	285	1
A4 v2	4	8.00	40	4	D1 v1	1	3.50	50	12
A8 v2	8	16.00	80	2	D2 v1	2	7.00	100	6
A2m v2	2	16.00	20	8	D3 v1	4	14.00	200	3
A4m v2	4	32.00	40	3	D4 v1	8	28.00	400	1
A8m v2	8	64.00	80	1	D1 v2*	1	3.50	50	14
D11 v1	2	14.00	100	4	D2 v2*	2	7.00	100	7
D12 v1	4	28.00	200	2	D3 v2*	4	14.00	200	3
D13 v1	8	56.00	400	1	D4 v2*	8	28.00	400	1
D11 v2*	2	14.00	100	5	F1*	1	2.00	16	16
D12 v2*	4	28.00	200	2	F2*	2	4.00	32	8
D13 v2*	8	56.00	400	1	F4*	4	8.00	64	4
G1*	2	28.00	384	1	F8*	8	16.00	128	2

(in GiB), and HD (in GB) columns. The Mach. column indicates the number of machines with this configuration that could be instantiated as slaves. This number is equal to either $\lfloor h/p \rfloor$ or $\lfloor 16/c \rfloor$, whichever is smaller, where h is the available budget per hour, p is the cost per hour of instantiating one machine, and c is the number of cores one machine has.

Azure groups similar machine configurations [Microsoft 2017]. In Table 3, a white backdrop indicates machines classified as "General Purpose—Balanced CPU to memory ratio." A light-gray backdrop indicates "Compute Optimized—High CPU to memory ratio" machines. Those on a dark-gray backdrop, finally, are "Memory Optimized—High memory to core ratio" machines. Configurations with an asterisk (*) by their denominations comprise machines that Azure offers with or without an SSD.

Results

All tests were executed on a set of 120 formulas (searchonmath.com/formulas, accessed: May 1, 2018) from [Salem et al. 1992, Stewart 2012, Kamali and Tompa 2013, Stalnaker and Zanibbi 2015, Pavan Kumar et al. 2012].

The overall testing scheme for each line of Table 3 (each configuration of the slave machines in Figure 1) was the following. The first of the 120 formulas was submitted for search to the master machine (of type A3 Basic), which then passed it on to the slave machines (of types dependent upon the configuration in question, as per Table 3) and awaited their results. Having received these, the master machine put together and sorted the final list of results and proceeded to submitting the second formula in the set. This was repeated until all 120 formulas were searched.

This full search pass over all 120 formulas was repeated 41 times for each of the configurations of Table 3. The time spent on each pass was recorded and, at the end,

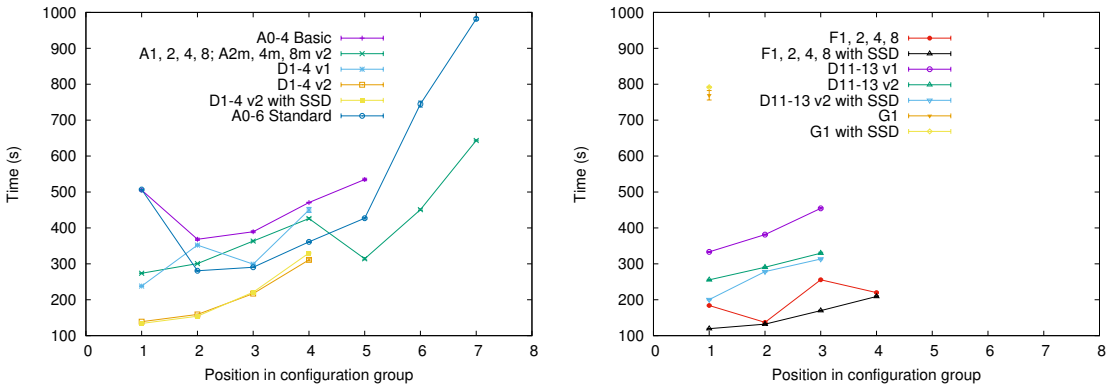


Figure 2. Time spent on General Purpose machines (left panel) and on Compute Optimized and Memory Optimized machines (right panel).

the average time of all 41 executions was found and its confidence interval estimated (at the 99% level). We note that each time measurement disregards every communication delay between the client and the master (cf. Figure 1). As a result, all time figures we report are search-related, referring to processing time at the master or at a slave, or to internal network delays of the distributed system. We give results in Figures 2(a) and 2(b), respectively for the machines of Azure type General Purpose and for those of the other two types (Compute Optimized and Memory Optimized).

Each plot in these figures refers to a group of slave-machine configurations, as implied by the horizontal rules in Table 3, and positions each of the group's configurations on the abscissa axis in the order given in the table. So, for example, configurations A0–4 Basic are grouped together, with A0 Basic appearing leftmost in Figure 2(a), followed by A1 Basic, and so on. It is also worth noting that the confidence intervals are often negligible and therefore hard to discern in the figures.

As it turns out, the best scenario for the SearchOnMath system is the slave-machine configuration F1 with SSD, which comprises 16 identical single-core machines, each with 2 GiB of RAM and a 16-GB SSD. With this configuration, the time needed to search for the 120 formulas was about 120 seconds on average (roughly 1 second per formula), with a confidence interval of approximately $\pm$ 0.86 seconds.

Notwithstanding this, we note that in general the SSD-based configurations did not result in a large difference when compared to their HD-based counterparts. This was expected, given that SearchOnMath carries the formulas in memory while running, thus considerably reducing the need for access to secondary storage. Two exceptions to this note occurred for configurations F1 and F4, in which case time differences were indeed significant. Nevertheless, we are unable to explain such differences on grounds of the SearchOnMath algorithms, and must therefore speculate that they have to do with factors internal to Azure.

Conclusions and Acknowledgments

Carrying out the experiments described in this paper has allowed us to observe the functioning of SearchOnMath on a variety of configurations of the Microsoft Azure cloud environment. We experimented with all configurations compatible with our BizSpark

status and, within these limits, identified a configuration capable of supporting 1-second searches for 120 (out of just over 1 900 000) formulas. At the relatively modest cost currently afforded us by the Microsoft BizSpark program, these experiments will help us envisage plans to scale up operations. We note, finally, that the 120 formulas selected for the experiments will remain available from SearchOnMath for possible future use in further comparative studies.

We thank the Federal University of Alfenas and NidusTec Business Incubator, as well as CNPq, CAPES, and a FAPERJ BBP grant for financial support. We also thank Microsoft for the opportunity to participate in their BizSpark program.

References

Asperti, A., Guidi, F., Coen, C. S., Tassi, E., and Zacchiroli, S. (2006). A content based mathematical search engine: Whelp. In *LNCS 3839*, pages 17–32. Springer.

Hu, X., Gao, L., Lin, X., Tang, Z., Lin, X., and Baker, J. B. (2013). Wikimirs: A mathematical information retrieval system for wikipedia. In *Proceedings of JCDL'13*, pages 11–20.

Kamali, S. and Tompa, F. W. (2013). Retrieving documents with mathematical content. In *Proceedings of SIGIR'13*, pages 353–362.

Kohlhase, M. and Sucan, I. (2006). A search engine for mathematical formulae. In *LNCS 4120*, pages 241–253. Springer.

Lin, X., Gao, L., Hu, X., Tang, Z., Xiao, Y., and Liu, X. (2014). A mathematics retrieval system for formulae in layout presentations. In *Proceedings of SIGIR'14*, pages 697–706.

Microsoft (2017). Azure price calculator. Accessed: Nov. 23, 2016.

Pavan Kumar, P., Agarwal, A., and Bhagvati, C. (2012). A structure based approach for mathematical expression retrieval. In *LNCS 7694*, pages 23–34. Springer.

Salem, L., Testard, F., and Salem, C. (1992). *The Most Beautiful Mathematical Formulas*. Wiley.

Schellenberg, T., Yuan, B., and Zanibbi, R. (2012). Layout-based substitution tree indexing and retrieval for mathematical expressions. In *Proceedings of SPIE 8297*, page 829701. SPIE.

Schubotz, M., Grigorev, A., Leich, M., Cohl, H. S., Meuschke, N., Gipp, B., Youssef, A. S., and Markl, V. (2016). Semantification of identifiers in mathematics for better math information retrieval. In *Proceedings of SIGIR'16*, pages 135–144.

Stalnaker, D. and Zanibbi, R. (2015). Math expression retrieval using an inverted index over symbol pairs. In *Proceedings of SPIE 9402*, page 940207. SPIE.

Stewart, I. (2012). *In Pursuit of the Unknown: 17 Equations That Changed the World*. Basic Books.

Zanibbi, R., Davila, K., Kane, A., and Tompa, F. W. (2016). Multi-stage math formula search: Using appearance-based similarity metrics at scale. In *Proceedings of SIGIR'16*, pages 145–154.

Anonimização de Streaming de Dados em DOCA

Bruno C. Leal[1], **Israel C. Vidal**[1], **Javam C. Machado**[1]

[1]Laboratório de Sistemas e Banco de Dados (LSBD)
Universidade Federal do Ceará (UFC) – Fortaleza – CE – Brasil

{bruno.leal,israel.vidal,javam.machado}@lsbd.ufc.br

Abstract. *Online scenarios are increasingly more common, providing great opportunities for data analysis. Such data usually contains sensitive information and should be anonymized to guarantee individuals' privacy. This work proposes DOCA, a differentially private approach for publishing data streaming in non-interactive scenarios using an online microaggregation strategy to obtain better utility.*

Resumo. *Cenários online são cada vez mais comuns, propiciando grandes oportunidades de análise de dados. Frequentemente, esses dados contém informações sensíveis de indivíduos e, portanto, devem ser anonimizados para garantir sua privacidade. Este trabalho propõe DOCA, uma abordagem diferencialmente privada para publicação de streaming de dados em cenários não interativos utilizando uma estratégia de microagregação online para obtenção de melhor utilidade.*

1. Introdução

A coleta de dados vem crescendo em grandes proporções nos últimos anos, tanto em volume quanto em velocidade. Esses dados apresentam valiosas oportunidades para análise, e.g. técnicas de aprendizado de máquina para classificação ou predição. Cenários *online*, tais como monitoramento e análises de ações da bolsa, análise de densidade de tráfego, *IoT* e *Smart cities*, estão se tornando cada vez mais comuns. Porém, grande parte das informações são oriundas de dados sensíveis dos indivíduos e devem ser tratadas com cuidado para garantir a privacidade dos mesmos.

Privacidade Diferencial (PD) [Dwork and Roth 2014] tem se tornado o modelo padrão de privacidade de dados em detrimento de outros modelos como k-anonimato [Sweeney 2002]. Sua aplicação se dá, principalmente, na proteção de informações estatísticas obtidas por consultas interativas. Neste cenário, o resultado das consultas em formato agregado, como por exemplo histogramas, limita as possibilidades de análises. Para oferecer maior capacidade de análise, a publicação do dado em cenário não interativo, numa versão anonimizada do dado original, é desejável [Soria-Comas et al. 2014]. Todavia, para a publicação de dados em cenário não interativo utilizando PD, a quantidade de ruído necessária para proteger cada item do conjunto de dados pode ser tão grande que o dado anonimizado gerado pode não ter utilidade.

A privacidade de indivíduos no contexto de *streaming* de dados é, portanto, extremamente relevante e deve ser atacada. Este trabalho propõe, então, a abordagem DOCA (***D**ifferential Privacy **O**nline **C**lustering and **A**nonimization*) para garantir a Privacidade Diferencial de forma a minimizar o ruído no contexto de publicação não interativa, garantindo maior liberdade de análise para uma *streaming* de dados numéricos.

2. Trabalhos Relacionados

Para a aplicação da Privacidade Diferencial em consultas que retornam dados numéricos (e.g. *COUNT*), o Mecanismo de Laplace [Dwork and Roth 2014] é o mais amplamente utilizado e aceito. Este gera um ruído a ser adicionado à resposta real com base em dois parâmetros: *(i)* o *budget* ϵ e *(ii)* a sensibilidade Δf. O primeiro é um parâmetro de entrada para ajustar o nível de privacidade desejado. Já a sensibilidade depende do domínio da consulta e é definida como o máximo impacto possível de um indivíduo na resposta. A sensibilidade é utilizada para garantir que a presença ou ausência de um indivíduo no dado não irá afetar substancialmente a resposta.

PD é adotada, principalmente, no contexto interativo para consultas de informações estatísticas [Chen et al. 2011, Xu et al. 2013]. Entretanto, trabalhos nesse contexto limitam as possibilidades de análises e não se relacionam diretamente com o tipo de publicação do DOCA. No contexto de publicação de versões anonimizadas no formato original do dado, i.e., versões não interativas, os trabalhos [Zhang et al. 2017, Bindschaedler et al. 2017] geram tuplas sintéticas a partir de modelos construídos com base no dado original. Porém, não cobrem o contexto de *streaming* de dados.

Os trabalhos [Soria-Comas et al. 2014, Soria-Comas and Domingo-Ferrer 2017] publicam uma versão diferencialmente privada do conjunto de dados microagregado. Os autores argumentam que a perda de informação pela microagregação, que consiste em substituir cada valor de um grupo pelo centroide, é compensada pela redução do ruído necessário para proteger o dado microagregado. O DOCA utiliza a sensibilidade para geração do ruído de um grupo definida em [Soria-Comas and Domingo-Ferrer 2017] como $\Delta c = \Delta f/|C|$, onde C é o grupo a ser publicado e Δf a sensibilidade. O ruído para o grupo a ser publicado é gerado a partir do mecanismo de Laplace com média zero e escala $\Delta c/\epsilon$. Entretanto, essa estratégia é voltada para a anonimização de um conjunto de dados estático, enquanto o DOCA realiza a anonimização sobre um fluxo de dados contínuo.

3. DOCA - *Differential Privacy Online Clustering and Anonymization*

DOCA é uma abordagem de publicação de *streaming* de dados numéricos univariados que aplica a Privacidade Diferencial em um contexto não interativo, ao mesmo tempo que diminui a sensibilidade na adição de ruído para, ao publicar, manter a utilidade do dado sem comprometer a garantia de privacidade dos indivíduos.

No contexto de atuação do DOCA, é necessário considerar características específicas de *streaming* de dados que impactam fortemente no processo de anonimização por Privacidade Diferencial, a saber [Silva et al. 2013]: (1) o dado é potencialmente ilimitado, o que impede sua representação por inteiro em memória; (2) o algoritmo de microagregação deve ser *online*; (3) cada registro de entrada está sujeito a uma restrição de tempo de processamento (*delay constraint*) entre sua entrada e saída.

A Figura 1 ilustra as duas etapas do DOCA: (i) agrupamento *online* e (ii) anonimização com microagregação e adição de ruído.

3.1. Agrupamento *Online*

A anonimização realizada pelo DOCA adiciona, além da perda de informação originada pela microagregação, o ruído advindo do mecanismo de Laplace [Dwork and Roth 2014]

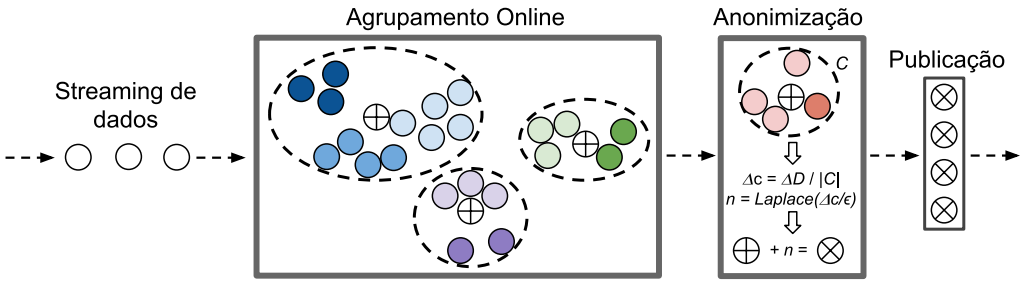


Figura 1. Uma visão geral da abordagem proposta.

para garantir a Privacidade Diferencial. Este ruído é diretamente proporcional à sensibilidade (Δf) e inversamente proporcional ao tamanho do grupo sobre o qual ele é aplicado pois a sensibilidade adotada no DOCA é $\Delta c = \Delta f/|C|$, onde C é o grupo a ser publicado.

Nós argumentamos que quanto maior o tamanho do grupo, menor quantidade de ruído se faz necessária, desde que o centroide seja representativo o suficiente para minimizar a perda de informação decorrente da microagregação. Então, para manter boa representatividade enquanto maximiza-se o tamanho dos grupos, criamos um novo algoritmo de clusterização online como uma variação da solução oferecida em [Cao et al. 2011]. Para garantir que os grupos tenham centroides representativos, utiliza-se um limiar para a máxima perda de informação de um grupo, dado por τ, que é continuamente atualizada (Algoritmo 1 linha 11) como a média da perda de informação dos últimos m grupos publicados. Então, dado que a representatividade de um grupo é garantida, esse grupo crescerá até que um de seus elementos atinja a *delay constraint*. Portanto, o agrupamento feito pelo DOCA no Algoritmo 2 nas linhas 3 a 5, onde a escolha do melhor cluster para um item (linha 4) é separado no Algoritmo 2 para melhor legibilidade, visa maximizar o tamanho dos grupos sem perder a representatividade dos centroides.

Algoritmo 1: DOCA

Entrada: $S, \Delta f, \beta, m, \epsilon$
Saída: Atualiza os clusters online C
/* S: streaming de dados */
/* Δf: sensibilidade global */
/* β: número máximo de clusters em memória */
/* m: número máximo de clusters usados para computar τ */
/* τ: média da perda de informação dos últimos m clusters publicados */
1 $C = [\,], I = [\,], \tau = 0, min_t = +\infty, max_t = -\infty$;
2 **para** *cada novo Registro* $r \in S$ **faça**
3 min_t = min(r, min_t); max_t = max(r, max_t);
4 Cluster c = BestSelection(r, C, τ, min_t, max_t);
5 adicione r em c;
6 Registro r_e = GetExpiringRecord();
7 **se** r_e != *Nulo* **então**
8 Cluster $c_e = \{c_i \in C | r_e \in c_i\}$;
9 i_{loss} = (max(c_e) - min(c_e)) / (max_t - min_t);
10 adicione i_{loss} em I;
11 τ = média dos últimos m valores $\in I$;
/* Passo 2-anonimização e publicação do cluster c_e */

Algoritmo 2: BESTSELECTION Seleciona o melhor grupo para um novo item

Entrada: r, C, τ, min_t, max_t
Saída: Cluster c_{best}

```
1  C_min = [ ], C_best = [ ], e_min = +∞;
2  para cada c ∈ C faça
3      e_i = GetEnlargment(r, c);
4      e_min = min(e_min, e_i);
5  para cada c ∈ C faça
6      se GetEnlargment(r, c) == e_min então
7          adicione c em C_min;
8          c_test = cópia de c; adicione r em c_test;
9          i_lossTest = (max(c_test) - min(c_test)) / (max_t - min_t);
10         se i_lossTest < τ então add c em C_best ;
11 se C_best é ∅ então
12     se tamanho de C < β então
13         c_new = novo Cluster; adicione c_new em C;
14         retorna c_new;
15     senão retorna um cluster de C_min com o menor tamanho ;
16 senão retorna um cluster de C_best com o menor tamanho ;
17
```

3.2. Anonimização

Essa etapa consiste em verificar se existe uma tupla que atingiu a *delay constraint* e, por isso, deve ser publicada (Algoritmo 1 linha 6). Para isso, publica-se o grupo ao qual essa tupla pertence, i.e., a tupla juntamente com as demais tuplas do mesmo grupo. Os passos de anonimização desse grupo que ocorrem após a linha 11 do Algoritmo 1 foram omitidos por questão de espaço e são descritos a seguir. Para publicação do grupo calcula-se o valor de seu centroide χ como a média dos valores. Após isso computa-se o valor do ruído η a ser inserido nesse grupo, dado por uma amostra aleatória obtida da distribuição de Laplace com média zero e escala $\Delta c/\epsilon$ (Δc definido na Subseção 3.1). Substitui-se, então, o valor de todas as tuplas por $\chi + \eta$ e, finalmente, publica-se o grupo anonimizado e remove-se o mesmo juntamente com seus elementos do conjunto de dados ainda em processamento e não publicados.

Observe que Δc não depende do tamanho do conjunto de dados, portanto pode ser diretamente aplicada no contexto de microagregação em *streaming* de dados. Além disso, como argumentado em [Soria-Comas and Domingo-Ferrer 2017], uma mesma amostra de ruído de Laplace deve ser gerada para todo o grupo. Caso contrário, se fosse gerado um ruído diferente para cada tupla, a quantidade de ruído gerada seria maior do que na abordagem tradicional, pois dessa forma teria-se, além do ruído da abordagem tradicional, a perda de informação decorrente do passo de microagregação. Perceba, também, que é satisfeita a restrição de limitação de memória, bem como a restrição de tempo de processamento até a saída, pois como o DOCA trata a chegada de uma nova tupla como uma unidade de tempo, jamais haverá mais tuplas em processamento do que o valor estabelecido para a restrição *delay constraint*.

4. Avaliação Preliminar

Para a avaliação experimental, foi simulada uma *streaming* de dados a partir do conjunto de dado BackBlaze [Backblaze 2017] no qual utilizamos o atributo *smart_9_raw*. O con-

junto de dado apresenta as seguintes características: 1.989.462 registros, 42.762 valores distintos, valor mínimo 1, valor máximo 66.413 e *skew* 0,7365. O *skew* foi medido através do Coeficiente de Assimetria dado por $\frac{1}{n*s^3}\sum_{i=1}^{n}(X_i - \bar{X})^3$, onde n é o tamanho do conjunto de dado, s é o desvio padrão e $\bar{X}$ é a média. O experimento foi repetido cinco vezes para todos os itens e, em cada execução, o item a ser processado foi selecionado aleatoriamente uma única vez.

Para computar os resultados apresentados, calculou-se a média das cinco execuções. Este critério foi adotado pois, como o algoritmo é *online*, a ordem em que os dados chegam pode ser relevante para o resultado final. O número de execuções deve ser baixo para não viciar o resultado, pois como característica do mecanismo de Laplace, a média de várias execuções tende a aproximar o resultado do valor real. Em relação aos parâmetros de entrada do DOCA foram utilizados os seguintes valores: *delay constraint* = 1000, $\beta = 50$, $m = 100$, escolhidos arbitrariamente e podem ser variados para buscar melhor utilidade; $\epsilon = 1,0$ pois é um valor aceito na literatura capaz de garantir a privacidade dos indivíduos e ainda manter bom nível de utilidade; sensibilidade $\Delta f = 99.618,0$, conforme utilizado por [Soria-Comas and Domingo-Ferrer 2017] onde $\Delta f = 1,5 * |max(conjuntodedado) - min(conjuntodedado)|$.

Para análise de utilidade comparou-se o DOCA com a estratégia ingênua de publicar os dados sem implementar o passo de microagregação *online*, simplesmente adicionando uma amostra do ruído de Laplace com média zero e escala $\Delta f/\epsilon$ a cada tupla. Avalia-se dessa forma pois, de acordo com nosso conhecimento, não há nenhum trabalho até o momento que trate da publicação diferencialmente privada de uma *streaming* de dados no contexto não interativo. Para essa avaliação utilizou-se o Erro Quadrático Médio (EQM) para representar a diferença numérica entre os valores reais e anonimizados. Os resultados são exibidos na Tabela 1.

Tabela 1. Avaliação de EQM entre DOCA e Abordagem ingênua.

EQM da abordagem ingênua	EQM do DOCA	Diminuição do EQM(%)
20.297.943.835,9597	143.064.439,6199	99,2952

Já para mostrar o quanto o dado anonimizado com o DOCA representa o dado original, mede-se o percentual de interseção entre as distribuições. A Figura 2 apresenta visualmente a comparação entre a distribuição original e a versão anonimizada através de um histograma com cem *bins* para uma das execuções. A média da área de interseção das cinco execuções obtida foi de $85,98\%$.

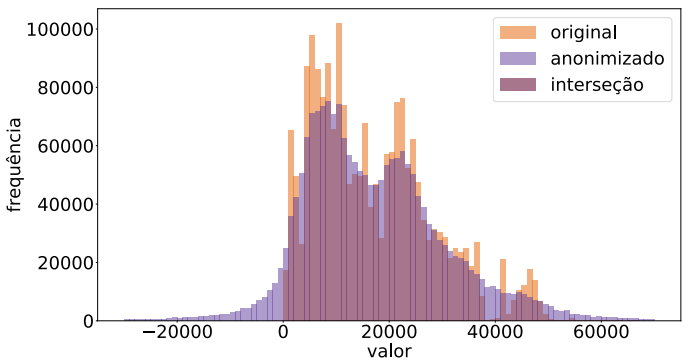


Figura 2. Distribuição original x anonimizada. Interseção = 86,01%.

5. Conclusão

Este artigo apresentou a estratégia DOCA para a anonimização de *streaming* de dados para publicação, fazendo uso de adição de ruído laplaciano nos agrupamentos desses dados. Os resultados preliminares comprovaram uma significativa redução de ruído adicionado, ao mesmo tempo que mantiveram um alto nível de privacidade, preservando ainda as características do conjunto de dados original. Várias oportunidades de trabalhos futuros são identificadas a partir dos nossos resultados. Dentre elas, podemos enumerar: (i) adaptar o DOCA para suportar *streaming* de tuplas com múltiplos atributos; (ii) definição de um modelo para ajuste dos parâmetros da fase de agrupamento *online*; (iii) suporte a atributos categóricos; e, talvez o mais importante, (iv) suporte a múltiplas tuplas de um mesmo indivíduo.

Referências

Bindschaedler, V., Shokri, R., and Gunter, C. A. (2017). Plausible deniability for privacy-preserving data synthesis. *Proc. VLDB Endow.*, 10(5):481–492.

Cao, J., Carminati, B., Ferrari, E., and Tan, K. L. (2011). Castle: Continuously anonymizing data streams. *IEEE Transactions on Dependable and Secure Computing*, 8(3):337–352.

Chen, R., Mohammed, N., Fung, B. C. M., Desai, B. C., and Xiong, L. (2011). Publishing set-valued data via differential privacy. *PVLDB*, 4(11):1087–1098.

Dwork, C. and Roth, A. (2014). The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3–4):211–407.

Backblaze (2017). The raw hard drive test data from 2017-01-01 to 2017-01-31. Online at `https://www.backblaze.com/b2/hard-drive-test-data.html`. acessed 2018-04-22.

Silva, J. A., Faria, E. R., Barros, R. C., Hruschka, E. R., Carvalho, A. C. P. L. F. d., and Gama, J. a. (2013). Data stream clustering: A survey. *ACM Comput. Surv.*, 46(1):13:1–13:31.

Soria-Comas, J. and Domingo-Ferrer, J. (2017). Differentially private data sets based on microaggregation and record perturbation. In *MDAI 2017, Kitakyushu, Japan, October, 2017, Proceedings*, pages 119–131.

Soria-Comas, J., Domingo-Ferrer, J., Sánchez, D., and Martínez, S. (2014). Enhancing data utility in differential privacy via microaggregation-based k-anonymity. *The VLDB Journal*, 23(5):771–794.

Sweeney, L. (2002). k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570.

Xu, J., Zhang, Z., Xiao, X., Yang, Y., Yu, G., and Winslett, M. (2013). Differentially private histogram publication. *The VLDB Journal*, 22(6):797–822.

Zhang, J., Cormode, G., Procopiuc, C. M., Srivastava, D., and Xiao, X. (2017). Privbayes: Private data release via bayesian networks. *ACM Trans. Database Syst.*, 42(4):25:1–25:41.

Um Estudo Comparativo entre Algoritmos de Proteção da Privacidade Aplicado à Bases de Dados na Área de Saúde

Francimaria Nascimento[1], Karliane Vale[2], Flavius Gorgônio[2]

[1]Departamento de Matemática e Informática Aplicada (DIMAP)
Universidade Federal do Rio Grande do Norte (UFRN)
Campus Universitário, Lagoa Nova, 59.078-970, Natal, RN, Brasil

francimariasantos@ufrn.edu.br

[2]Laboratório de Inteligência Computacional Aplicada a Negócios (LABICAN)
Universidade Federal do Rio Grande do Norte (UFRN)
Rua Joaquim Gregório, S/N, 59.300-000, Caicó, RN, Brasil

karliane@dct.ufrn.br, flavius@dct.ufrn.br

Abstract. *The growing increase in the volume of data which is collected, stored and shared by health institutions creates benefits for the process of decision making based on the knowledge obtained from applying data analysis and data mining techniques, aiming to achieve relevant information. Despite the obtained benefits, sharing this specific kind of data in its original raw format may compromise patients' privacy. In an attempt to validate solutions for this problem, this article considers and compares data anonymization and perturbation techniques, assessing their efficiency in providing privacy and safety of shared data, more specifically, when applied to databases of the health field.*

Resumo. *O crescente aumento no volume de dados coletados, armazenados e compartilhados por instituições da área de saúde gera benefícios para o processo de tomada de decisão com base no conhecimento adquirido a partir da aplicação de técnicas de análise e mineração de dados na extração de informações úteis. A despeito dos benefícios propiciados, o compartilhamento desses dados em seu formato original pode pôr em risco a privacidade dos pacientes. Na tentativa de validar soluções para este problema, este artigo compara algumas técnicas de anonimização e perturbação de dados, avaliando a eficácia dessas técnicas na garantia da privacidade e segurança de dados compartilhados, em particular, quando aplicadas a bases de dados na área de saúde.*

1. Introdução

Há uma crescente adoção de práticas de Tecnologia da Informação em pesquisas na área de saúde, o que resulta na coleta, armazenamento e compartilhamento de grandes volumes de dados, nos quais, a aplicação de técnicas de mineração de dados pode possibilitar a descoberta de informações potencialmente úteis. Entretanto, algumas instituições que realizam pesquisas na área de saúde e que necessitam fazer uso de várias fontes de dados, podem hesitar em compartilhar os dados entre si, pois seus registros normalmente possuem dados altamente sensíveis que não podem ser expostos, como por exemplo, informações pessoais dos pacientes [Kumari et al. 2012]. Por isso, nesses casos é importante compartilhar os dados, porém protegendo a privacidade dos registros.

O termo Preservação da Privacidade na Mineração de Dados (*Privacy Preserving Data Mining*) foi introduzido quase simultaneamente em [Agrawal and Srikant 2000] e [Lidell and Pinkas 2000]. Desde então, o assunto vem sendo amplamente estudado e é de grande relevância para a área de mineração de dados, dada a necessidade de algumas organizações extraírem o conhecimento existente em bases de dados compartilhadas, de forma colaborativa, porém com a garantia da proteção da privacidade.

Na literatura, estão descritas várias técnicas para garantir a proteção da privacidade dos dados, dentre elas, a Anonimização [Emam 2006, Byun et al. 2007] e a Perturbação [Agrawal and Srikant 2000], que são comparadas neste trabalho. Assim, faz-se necessário analisar a eficácia das técnicas utilizadas tanto com respeito à garantia da privacidade dos dados, quanto em relação ao nível de precisão dos resultados obtidos após a aplicação das técnicas. Pois, apesar das limitações e consequências, a utilização de técnicas para a manutenibilidade da privacidade dos dados é importante e deve ser aplicada quando se compartilham dados entre instituições a fim de se garantir a proteção dos registros. A não utilização dessas técnicas pode aumentar a vulnerabilidade dos dados, que podem vir a ser usados por terceiros mal-intencionados e causar diversos danos – problemas jurídicos, perda de privacidade, de patrimônio intelectual, entre outros – ao proprietário dos dados e/ou à instituição que os coletou.

2. Técnicas de Anonimização

Técnicas de anonimização realizam a remoção e/ou ofuscação de dados que possam auxiliar na identificação de um indivíduo em particular dentro de um conjunto de dados [Emam 2006]. Na literatura, estão descritos alguns modelos de anonimização, entre eles, estão o *k-anonymity* [Byun et al. 2007] e o *l-diversity* [Machanavajjhala et al. 2007]. O modelo *k-anonymity* tem como objetivo evitar que sejam feitas ligações entre atributos que identifiquem o proprietário do registro, desta forma, esse modelo exige que qualquer registro seja indistinguível de, pelo menos, $k - 1$ outros registros que possuam quasi-identificadores predeterminados (p.ex.: sexo, data de nascimento e CEP que, combinados, podem identificar o proprietário do registro).

Contudo, a homogeneidade de alguns valores sensíveis dentro de um conjunto de dados gera alguns problemas na proteção da privacidade com o modelo *k-anonymity*, pois a proteção dos *k*-indivíduos talvez não corresponda a todos os atributos sensíveis (ex.: diagnóstico de uma doença). Tendo em vista este problema, o modelo de *l-diversity* foi projetado para lidar com alguns problemas do modelo *k-anonymity* [Sinha and Kumar 2010].

O modelo *l-diversity* é baseado na premissa de que um conjunto de dados deve possuir pelo menos *l* atributos sensíveis "bem representados", para que a privacidade dos dados seja protegida [Machanavajjhala et al. 2007]. Algumas interpretações para o termo "bem representado" são demonstrados através dos seguintes princípios:

1 - Entropia - O cálculo da entropia *l-diversity* pela Equação 1, é usado para captar o número de grupos de atributos "bem representados", devido ao fato da entropia aumentar quando uma frequência se torna mais uniforme.

$$Entropia(E) = -\sum_{s \epsilon S} p(E,s) \log p_{(E,s)} \geq \log l \quad (1)$$

onde *E* representa uma classe de equivalência, s é o domínio de atributos sensíveis, e $p(E, s)$ é uma fração de registros em E que possuem atributos sensíveis s. Uma tabela é *l-diversity* se todas as classes de equivalência E, possuírem Entropia(E) $\geq \log l$.

Recursividade - Tem como finalidade certificar que o valor menos frequente não apareça raramente e que um valor muito frequente não apareça com muita frequência. Deste modo, dado um valor constante v, uma classe E satisfaz o princípio da recursividade se $r_i < v(r_l + r_{(l+1)} + \cdots + r_m)$, onde cada r representa o valor de um registro. Uma tabela possui recursividade *(v, l)-diversity*, se todas as classes também tiverem.

Para atingir o objetivo do *l-diversity*, um algoritmo denominado *Anatomize* foi proposto por [Xiao and Tao 2006]. Especificamente, este algoritmo separa os dados em duas tabelas, uma contendo os valores quasi-identificadores (QIT), que combinados podem identificar o usuário, e uma tabela sensível (ST), na qual são armazenados os atributos sensíveis.

3. Técnicas de Perturbação

Técnicas de perturbação adicionam ruídos aleatórios aos registros antes da etapa de mineração de dados, de forma que os resultados obtidos com os dados perturbados sejam aproximadamente os mesmos dos dados originais, possibilitando a extração de informações a partir da aplicação de técnicas de mineração de dados [Kedar et al. 2013, Sinha and Kumar 2010].

Com o objetivo de proteger dados de pesquisas médicas, [Liu et al. 2012] propuseram dois algoritmos para adicionar perturbações nas bases de dados em que são aplicadas técnicas de análise de agrupamento: i) Distância Aleatória no Domínio da Distância (*Random Distance in Distance Domain* - RDD), onde os registros primeiramente são agrupados com o algoritmo *k-means* e, em seguida, a cada registro é adicionado um ruído de uma distribuição Gaussiana, aplicando pequenos ajustes nas distâncias entre o dado e o centroide do grupo ao qual ele foi classificado com o *k-means*, com a finalidade de manter os registros no mesmo grupo antes e depois de serem perturbados; e ii) Rotação em Torno do Centro de Agrupamento (*Rotation Around the Center of Clustering* - RACC), que distintamente de alguns algoritmos utilizados para adicionar perturbação à bases de dados, não perturba o registro adicionando diretamente um ruído aos dados originais, mas por uma pequena medida de distância aleatório $d_j (0 < d_j \leq 1)$ e um ruído aleatório $\theta(\theta_1, \theta_2, ..., \theta_n)$, onde, $\theta \, \epsilon \, (0, 2\pi)$. O algoritmo RACC, calcula o valor do registro perturbado Q com base nas Equações 2 e 3:

$$r = d_j \times dis(R, C) \tag{2}$$

$$\begin{bmatrix} Q_1 \\ Q_2 \\ \vdots \\ Q_n \end{bmatrix} = \begin{bmatrix} C_{i1} \\ C_{i2} \\ \vdots \\ C_{in} \end{bmatrix} + r \begin{bmatrix} \cos(\theta_1) \\ \cos(\theta_2) \\ \vdots \\ \cos(\theta_n) \end{bmatrix}, 0 < \theta < 2\pi \tag{3}$$

onde, r é resultante do produto da proporção de distância aleatória d_j e $dis(R, C)$, que é a distância euclidiana do registro R ao centróide C do agrupamento ao qual R faz parte.

3.1. Metodologia dos Experimentos

Para realização dos experimentos foram selecionadas 5 bases de dados da área de saúde disponíveis no UCI *Repository of Machine Learning Databases*: *Breast Cancer Wisconsin*, com 699 instâncias e 10 atributos; *Fertility Data Set*, com 100 instâncias e 10 atributos; *Lung Cancer Data Set*, com 32 intâncias e 56 atributos; *Mammographic Mass Data Set*, com 961 instâncias e 6 atributos; e *SPECTF Heart Data*, com 267 instâncias e 44 atributos. Para fins de simplificação foram renomeadas, respectivamente, como: Base de Dados 1, Base de Dados 2, Base de Dados 3, Base de Dados 4 e Base de Dados 5.

As técnicas de anonimização e perturbação foram escolhidas por serem bastante utilizadas na proteção da privacidade de dados na área de saúde [Emam 2006, Liu et al. 2012]. Dentre os modelos citados de anonimização, foi utilizado o *l-diversity*, pois, segundo [Li et al. 2007], o modelo de *k-anonymity* não é suficiente para a proteção dos dados. O modelo *l-diversity* foi aplicado com $l = 2$ (diversidade igual a dois), porque o atributo sensível escolhido foi o atributo *classe* que em todas as bases de dados possui apenas dois possíveis valores.

Dentre os algoritmos citados que implementam a estratégia proposta pela técnica de perturbação, foi utilizado o algoritmo RACC, pois, segundo [Liu et al. 2012], com o aumento no nível de perturbação o algoritmo RDD diminui o nível de precisão do resultado da mineração de dados, entretanto, o algoritmo RACC mantém essa precisão estável. O algoritmo de análise de agrupamentos escolhido foi o *k-means*, com *k* igual a 3. A escolha do *k-means* para a etapa de análise de agrupamentos se deu por este ser um dos algoritmos de agrupamento mais conhecidos e ser amplamente usado, além de simples e de fácil implementação [Oliveira and Zaïane 2007].

3.2. Quantificação de Erros de Agrupamento

A Equação 4 foi utilizada para medir, em valores percentuais, a taxa de erros de agrupamento, denotada por E_c, que deveria ser a menor possível:

$$E_c = \frac{1}{N} \sum_{i=1}^{k} ((|Grupo_i(D)|) - (|Grupo_i(D')|)) \quad (4)$$

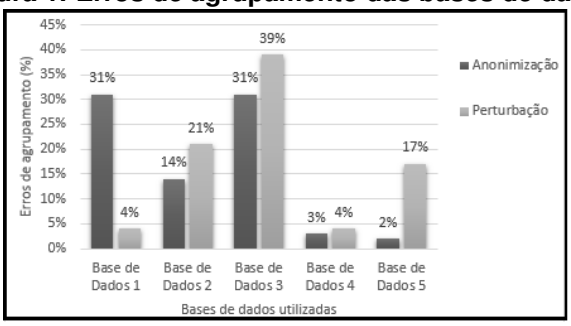
onde N representa o número de registros na base de dados original, k é o número de agrupamentos analisados, $|Grupo_i(D)|$ representa os registros do agrupamento dos dados originais e $|Grupo_i(D')|$ os registros do agrupamento dos dados distorcidos.

A Equação 4, proposta por [Oliveira and Zaïane 2010], foi escolhida por não considerar simplesmente a quantidade de pontos que cada agrupamento possui. Ao invés disso, é levado em consideração o agrupamento real de cada ponto, pois são comparados os rótulos dos agrupamentos de cada ponto antes e depois da distorção. Na Figura 1 são descritos os resultados obtidos no experimento que compara as técnicas de anonimização e perturbação, tendo sido utilizada a taxa de erro para medir a eficácia do algoritmo.

3.3. Quantificação da Privacidade

Além da medida de erros de agrupamento, [Oliveira and Zaïane 2010] propuseram quantificar a privacidade dos dados a partir de uma medida de segurança Sec, com base em uma medida de variância dada por $var(x - x')$, onde x representa um atributo original e x' o atributo distorcido, de modo que quanto maior o valor de $var(x - x')$ melhor o

Figura 1. Erros de agrupamento das bases de dados.



resultado. Esta medida pode ser expressa em uma escala invariante, no que diz respeito à variação de valores da variável original, descrita na Equação 5, onde quanto maior a variância melhor o resultado obtido.

$$Sec = \frac{var(x - x')}{var(x)} \tag{5}$$

Os resultados obtidos pela aplicação da técnica de perturbação nas bases de dados citadas são apresentados na Tabela 1, onde estão descritos o valor mínimo (V_{min}), o valor máximo (V_{max}), a média ($V_{méd}$) e o desvio padrão (σ), calculados a partir da Equação 5.

Tabela 1. Nível de privacidade do dados perturbados (%)

Bases de Dados	Vmin	Vmax	Vméd	Vσ
Base de Dados 1	0,97	1,82	1,15	0,22
Base de Dados 2	1,35	2,91	2,16	0,42
Base de Dados 3	4,38	11,41	7,44	1,51
Base de Dados 4	0,61	1,60	0,88	0,32
Base de Dados 5	5,05	8,13	6,67	0,65

A partir dos testes realizados, foi possível verificar que em todas as bases de dados em que foi aplicada a técnica de anonimização os resultados obtidos pela aplicação da Equação 5 foram iguais a zero. Isso acontece, como citado anteriormente, em função do algoritmo de anonimização utilizado não realizar distorções nos atributos dos registros, e sim, criar regras pelas quais mais de um registro possuam atributo sensível distinto e os atributos quasi-identificadores semelhantes, deste modo alguns registros que não se encaixam nas regras geradas são suprimidos da base de dados [Machanavajjhala et al. 2007].

Desta forma, a probabilidade de descobrir o valor do atributo sensível é $\frac{1}{l}$, onde l é a diversidade que se deseja alcançar. Assim, quanto maior o valor de l, menor a probabilidade de descoberta do atributo sensível. Por exemplo, uma base de dados que possui apenas dois possíveis valores para um atributo sensível, poderá ter no máximo $l = 2$, com isso o valor do atributo sensível pode ser reconstruído com 50% de probabilidade.

4. Conclusões e Trabalhos Futuros

A partir dos resultados obtidos, foi possível inferir que o algoritmo aplicado para perturbação dos dados mantém um certo *trade-off* entre privacidade dos dados e precisão dos resultados da análise de agrupamentos, ou seja, mais privacidade implica diretamente

em menos precisão. Também foi possível constatar que as bases de dados que possuem maiores dimensionalidades possuem melhores resultados quanto à privacidade dos dados.

Quanto à eficácia, verificou-se mais erros de agrupamentos nas bases de dados em que foi aplicada a técnica de perturbação, sendo que estes ocorrem devido à distribuição espacial dos dados, pois o algoritmo de perturbação utilizado rotaciona os dados em torno do centroide de cada agrupamento gerado pelo algoritmo. Desta forma, quando os agrupamentos são próximos, alguns registros ficam próximos do centroide do agrupamento vizinho, gerando assim erros de agrupamento.

Como os resultados obtidos nesta pesquisa são iniciais, é importante considerar como proposta de trabalho futuro uma análise mais profunda nas técnicas analisadas, averiguando diferentes configurações, como por exemplo modificando quantidade e tipos de atributos sensíveis, além da quantidade de agrupamentos usados.

Referências

Agrawal, R. and Srikant, R. (2000). Privacy-preserving data mining. In *ACM Sigmod Record*, volume 29, pages 439–450, California. ACM.

Byun, J., Kamara, A., Bertino, E., and Li, N. (2007). Efficient k-anonymization using clustering techniques. In *12th Int Con Database Syst Adv App*, pages 188–200, Berlin.

Emam, K. (2006). *Data anonymization practices in clinical research: a descriptive study*. CHEO Research Institute, Ottawa.

Kedar, S., Dhawale, S., Vaibhav, W., Kadam, P., Wani, S., and Ingale, P. (2013). Privacy preserving data mining. *Advanced Res. in Comp. and Com. Eng.*, 2(4):1677–1680.

Kumari, A., Rao, R., and Suman, M. (2012). Vector quantization for privacy preserving clustering in data mining. *Advance Computing*, 3(6):69–74.

Li, N., Li, T., and Venkatasubramaniam, S. (2007). t-Closeness. In *23rd International Conference on Data Engineering, 2007*, pages 106–115, Istambul. IEEE.

Lidell, Y. and Pinkas, B. (2000). Privacy-preserving data mining. In *International Cryptology Conference on Advances in Cryptology, 1880*, pages 36–54, California. Springer.

Liu, L., Yang, K., Hu, L., and Li, L. (2012). Using noise addition method based on pre-mining to protect healthcare privacy. *Journal Control Eng. App. Inf.*, 14(2):58–64.

Machanavajjhala, A., Kifer, D., Gehrke, J., and Venkitasubramaniam, M. (2007). l-diversity: Privacy beyond k-anonymity. *ACM Transactions on KDD*, 1(3):52.

Oliveira, S. and Zaïane, O. (2007). A privacy-preserving clustering approach toward secure and effective data analysis for business collaboration. *Com&Sec*, 26(1):81–93.

Oliveira, S. and Zaïane, O. (2010). Privacy Preserving Clustering by Data Transformation. *Journal of Information and Data Management*, 1(1):37–51.

Sinha, B. and Kumar, J. (2010). *Privacy Preserving Clustering In Data Mining*. PhD thesis, National Institute of Technology Rourkela, Rourkela.

Xiao, X. and Tao, Y. (2006). Anatomy: Simple and effective privacy preservation. In *Proc. of the 32nd Int. Conf. on Very Large Data Bases*, pages 139–150, Hong Kong.

SBBD 2018

Tutorials

In-Memory Analytic DBMSs: Design and Lessons Learned

Pedro Eugenio Rocha Pedreira

Facebook Inc.
1 Hacker Way
Menlo Park, CA, USA
pedroerp@fb.com

Abstract

In-memory DBMSs are systems that primarily rely on main memory for data storage. Many open source and commercial in-memory OLTP DBMSs were developed and studied in the last decade, such as VoltDB, Oracle TimesTen and Hekaton, as the amount of main memory available outpaced the size of most transactional working sets. As memory density kept increasing, a more recent trend is to leverage in-memory systems to speedup critical analytical workloads. In this scenario, in-memory analytic DBMSs have become a viable option to speedup analytic queries to important datasets.

Today, there are three main configurations in which in-memory analytic DBMSs can be used. First, these systems can be seen as *accelerators* for critical data marts which are part of much larger – and slower – data warehouses. These use cases focus on fast reporting and allowing users to interactively slice and dice portions of the data warehouse data. Second, these systems can be used to incrementally consume logs from messaging systems such as Kafka, providing analytics over realtime data and minimizing data latency. Third, some systems like MemSQL, SAP Hana and Hyper take a different set of trade-offs and are able to handle both OLTP and OLAP workloads in a hybrid configuration, also known as HTAP (Hybrid Transactional/Analytic Processing).

Besides the differences from traditional disk based analytic systems, in-memory analytic DBMSs also differ from in-memory OLTP DBMSs in many aspects. Since analytic queries are mostly composed of large memory scans, the lack of data locality, instruction and data cache misses, virtual and non-inlined function calls, remote NUMA accesses and branch mispredictions all have a non-trivial impact on query performance. JIT compilation at query-time, therefore, becomes a first class citizen to tighten scan loops, as opposed to the traditional approach based on stored procedures. In addition, the traditional *volcano* iterator chaining model, which is usually implemented using virtual calls, becomes less suited for these types of system and more data-centric query processing models are developed.

Furthermore, since analytic systems are designed to store large volumes of data, in-memory analytic DBMSs must carefully choose which parts of the dataset to keep in memory in order to minimize I/O reads at query time. The data structures used to store and index the in-memory data can also be substantially different from the ones traditionally used in clustered and secondary indexes in order to provide fast and cache-friendly scans as well as some form of data pruning. Lastly, transactions on these systems might also have different requirements due to the low number of transactions per second and high ingestion rates, creating opportunity for the development of novel and more lightweight concurrency control protocols.

This tutorial will discuss how in-memory analytic DBMSs are designed and built and outline the architecture of some state-of-art in-memory database systems, stressing the characteristics that differentiate them from the traditional DBMS design literature. In addition, the author will discuss some of the lessons learned while building and providing Cubrick as a service at Facebook, and highlight some of the many research opportunity avenues.

Author

Pedro Eugenio Rocha Pedreira is a Software Engineer at Facebook focused on database research. For the last 5 years he has led the team that develops Cubrick, a new in-memory analytic DBMS that targets interactive queries over highly dynamic datasets. Cubrick powers many internal analytic products and workloads and leverages a new indexing technique called Granular Partitioning, which he has proposed in his Ph.D thesis. Prior to joining Facebook, Pedro spent about 4 years working for the Brazilian Government, supporting the databases and infrastructure leveraged by the Brazilian Electoral Systems. He received his B.Sc., M.Sc and Ph.D from the Federal University of Parana (UFPR) in Curitiba/PR, Brazil.

References

Chen, J., Jindel, S., Walzer, R., Sen, R., Jimsheleishvilli, N., and Andrews, M. (2016). The MemSQL Query Optimizer: A modern optimizer for real-time analytics in a distributed database. *Proc. VLDB Endow.*, 9(13):1401–1412.

Diaconu, C., Freedman, C., Ismert, E., Larson, P.-A., Mittal, P., Stonecipher, R., Verma, N., and Zwilling, M. (2013). Hekaton: Sql server's memory-optimized oltp engine. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, SIGMOD'13, pages 1243–1254, New York, NY, USA. ACM.

Färber, F., Cha, S. K., Primsch, J., Bornhövd, C., Sigg, S., and Lehner, W. (2012). SAP HANA Database: Data Management for Modern Business Applications. *SIGMOD Rec.*, 40(4):45–51.

Kemper, A. and Neumann, T. (2011). HyPer: A Hybrid OLTP&OLAP Main Memory Database System Based on Virtual Memory Snapshots. In *Proceedings of the 2011 IEEE 27th International Conference on Data Engineering*, ICDE '11, pages 195–206, Washington, DC, USA. IEEE Computer Society.

Kohn, A., Leis, V., and Neumann, T. (2018). Adaptive execution of compiled queries. In *Proceedings of the 2018 IEEE 34th International Conference on Data Engineering*, ICDE'18, Paris, France. IEEE Computer Society.

Mukherjee, N., Chavan, S., Colgan, M., Das, D., Gleeson, M., Hase, S., Holloway, A., Jin, H., Kamp, J., Kulkarni, K., Lahiri, T., Loaiza, J., Macnaughton, N., Marwah, V., Mullick, A., Witkowski, A., Yan, J., and Zait, M. (2015). Distributed architecture of oracle database in-memory. *Proc. VLDB Endow.*, 8(12):1630–1641.

Neumann, T. (2011). Efficiently compiling efficient query plans for modern hardware. *Proc. VLDB Endow.*, 4(9):539–550.

Pedreira, P., Croswhite, C., and Bona, L. (2016). Cubrick: Indexing millions of records per second for interactive analytics. *Proc. VLDB Endowment*, 9(13):1305–1316.

Pedreira, P., Lu, Y., Pershin, S., Dutta, A., and Croswhite, C. (2018). Rethinking concurrency control for in-memory olap dbmss. In *Proceedings of the 2018 IEEE 34th International Conference on Data Engineering*, ICDE'18, Paris, France. IEEE Computer Society.

Coleta, Integração e Pré-processamento de Dados de Múltiplas Fontes

Natércia A. Batista[1], Michele A. Brandão[1], Michele Brito[1], Daniel H. Dalip[2], Mirella M. Moro[1]

[1]Universidade Federal de Minas Gerais – Belo Horizonte – Brasil

[2]Centro Federal de Educação Tecnológica de Minas Gerais – Belo Horizonte – Brasil

{natercia,micheleabrandao,mibrito,mirella}@dcc.ufmg.br,
hasan@decom.cefetmg.br

Resumo. *Dados extraídos da Web são cada vez mais heterogêneos e não estruturados, representando desafios para atividades de coleta, integração e pré-processamento de dados. Existem estudos que são "orientados a dados", i.e., com base nos dados disponíveis, mas seus resultados ficam restritos aos respectivos dados. Em contraponto, vários problemas existem antes de se identificar quais dados são necessários para solucioná-los, e muitas vezes, são necessários dados de múltiplas fontes. Nesse contexto, o primeiro problema ao lidar com dados provenientes da Web é definir a estratégia de coleta, que pode ser classificada de acordo com o período e a forma de buscar a semente. Outro problema é definir uma estratégia para integrar os dados de diferentes fontes de forma a ter uma visão uniforme para usuários ou aplicações, além de armazená-los de maneira a permitir uma consulta eficiente. Finalmente, pode ser necessário realizar o pré-processamento de dados, que acontece antes ou depois da integração de dados, e envolve resolver dados faltantes e duplicados, normalização, etc. Este tutorial aborda esses três problemas de forma integrada com foco em questões práticas e de pesquisa.*

1. **Coleta de Dados.** Esta parte do tutorial cobre as principais estratégias de coleta de múltiplas fontes, bem como os principais desafios. Especificamente: uma visão geral sobre os três principais tipos de coleta considerando o período de realização da mesma (contínua, periódica e/ou ocasional) e a busca da semente -- entidade alvo de coleta ou ponto inicial (busca em largura, caminhamento aleatório e caminhamento aleatório Metropolis-Hastings [Gjoka et al. 2010]). Ademais, são apresentados exemplos práticos e desafios para cada estratégia.
2. **Estratégias para Integrar Dados de Múltiplas Fontes.** Após os dados serem coletados de múltiplas fontes, eles precisam ser integrados. Note que é possível coletar dados de cada fonte e armazená-los de forma separada para posterior integração, ou já armazenar todos os dados em um único local de forma integrada à medida que cada coleta é realizada. Nesta etapa, são abordadas vantagens e desvantagens dessas duas estratégias e das diferentes formas de armazenamento (planilhas, arquivos CSV, banco de dados relacionais, etc).

3. **Pré-processamento de Dados.** Nesta etapa do tutorial são abordados problemas geralmente encontrados nos dados após a realização da coleta: (*i*) valores faltantes - quando nenhum valor é armazenado para uma variável; (*ii*) veracidade dos dados - refere-se a viés, anormalidades e ruídos presentes nos dados; (*iii*) remoção de dados duplicados - dados iguais inseridos múltiplas vezes; (*iv*) falta de normalização - processo de reestruturar os dados a uma forma comum; e (*v*) redução dos dados - processo de minimizar a quantidade de dados a serem armazenados.
4. **Aplicações Reais.** Existem diferentes estudos que combinam múltiplas fontes de dados com objetivos distintos [Batista et al. 2017, Brandão et al. 2017, Dalip et al. 2013, Farnadi et al. 2018]. Por exemplo, Batista et al. (2017) medem a força dos relacionamentos entre desenvolvedores através da combinação de dados originados de um banco de dados relacional disponível na Web e dados coletados utilizando a API (*Application Programming Interface*) do GitHub. Brandão et al. (2017) utilizam dados de múltiplas fontes (DBLP e páginas Web) para fornecer visualizações com informações mais completas sobre pesquisadores. Ademais, Dalip et al. (2013) geram um ranking de respostas para perguntas no Stack Overflow por meio de indicadores. Dentre tais indicadores, utilizou-se a comparação entre os vocabulários das respostas com os vocabulários de bons artigos do Wikipedia. Todos esses exemplos de uso real da combinação de múltiplas fontes de dados são abordados nesta etapa mais prática do tutorial.
5. **Conclusão e Trabalhos Futuros.** Finalmente, além de abordar os principais conceitos, desafios e exemplos relacionados à coleta, integração e pré-processamento de dados de múltiplas fontes, discutimos problemas em aberto e possíveis trabalhos futuros a fim de incentivar pesquisas relacionadas a este tutorial.

Referências

Batista, N. A., Brandão, M. A., Alves, G. B., Silva, A. P. C. da, and Moro, M. M. (2017) "Collaboration strength metrics and analyses on GitHub." In *WI*, pp. 170-178.

Brandão, M. A., Diniz, M. A., Sousa, G. A., Moro, M. M. (2017) "Visualizing Co-Authorship Social Networks and Collaboration Recommendations With CNARe." *Graph Theoretic Approaches for Analyzing Large-Scale Social Networks*: 173-188.

Dalip, D. H., Gonçalves, M. A., Cristo, M., and Calado, P. (2013) "Exploiting user feedback to learn to rank answers in q&a forums: a case study with stack overflow." In *SIGIR*, pp. 543-552.

Farnadi, G., Tang, J., Cock, M. D., and Moens, M-F. (2018) "User Profiling through Deep Multimodal Fusion." In *WSDM*, pp. 171-179.

Gjoka, M., Kurant, M., Butts, C. T., and Markopoulou, A. (2010) "Walking in facebook: A case study of unbiased sampling of osns." In *IEEE Infocom*, pp. 1-9.

SBBD 2018

Keynotes

SBBD – Para Que e Para Quem?

Sérgio Lifschitz

Depto. de Informática – Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio)
Rua Marquês de São Vicente 225 – 22.451-900 – Rio de Janeiro – RJ – Brazil

sergio@inf.puc-rio.br

Resumo. *Neste ano de 2018, o 33o SBBD acontece junto com o 44o VLDB. O primeiro é o maior evento acadêmico nacional, enquanto o último é um dos maiores e mais importantes da área no âmbito internacional. Durante o SBBD, a comunidade brasileira de bancos de dados tem a oportunidade de acompanhar de perto os principais avanços científicos e tecnológicos na área, além de avaliar a importância do evento e o papel dos professores e pesquisadores na formação de pessoal e na contribuição para a sociedade. Nesta palestra, pretendo revisitar alguns dos diversos temas de P&D a que venho me dedicando ao longo dos anos, desde bancos de dados dedutivos até a análise de dados, passando pelas aplicações na bioinformática e pelos SGBDs autônomos. Aproveitarei também o convite para provocar uma reflexão sobre a nossa grande área de pesquisa, com foco na relevância e no papel de eventos como o SBBD para a ciência e a tecnologia no país.*

Sérgio Lifschitz is an Associate Professor at the Informatics Department of Pontifical Catholic University of Rio de Janeiro (PUC-Rio), Brazil. Sérgio has obtained his doctor's degree at the École Nationale Supérieure des Télécommunications (now Télécom ParisTech), Paris, France (1994). He also holds an M.Sc.(1990) and a B.Sc. (1986), both in Electrical Engineering, from PUC-Rio. His primary research area involves database systems, including applications in bioinformatics, social networks data, and autonomic tuning and management. Sérgio often participates in SBBD events, either with tutorials (three times), eleven full papers published, other 18 contributions for SBBD co-located events, one best paper and two best demo prizes, more than thirty times a member of evaluation committees and once the SBBD program chair.

Fast, Real-time Analysis on All Kinds of Data

Anastasia Ailamaki

Ecole Polytechnique Fédérale de Lausanne (EPFL)
BC 226, Station 14 – CH-1015 – Lausanne – Switzerland

anastasia.ailamaki@epfl.ch

Abstract. *Today's scientific and business processes heavily depend on fast and accurate data analysis. Data scientists are routinely overwhelmed by the effort needed to manage the volumes of data produced. As general-purpose data management software is often inefficient, hard to manage, or too generic to serve today's applications, businesses increasingly turn to specialised data management software, which can only handle one data format, and then resort to data integration solutions. With the exponential growth of dataset size and complexity, however, data format-specific solutions no longer scale for efficient analysis, thereby slowing down the cycle of analysing and understanding the data, and making decisions. I will illustrate the different nature of problems we face when managing heterogeneous datasets, and how these translate to fundamental challenges for the data management community. Then I will introduce RAW, a new solution inspired by these challenges. RAW overturns long-standing assumptions, enables meaningful and timely results, and promotes timely discovery.*

Anastasia Ailamaki is a Professor of Computer and Communication Sciences at the Ecole Polytechnique Federale de Lausanne (EPFL) in Switzerland and the co-founder of RAW Labs SA, a swiss company developing real-time analytics infrastructures for heterogeneous big data. Her research interests are in data-intensive systems and applications, and in particular (a) in strengthening the interaction between the database software and emerging hardware and I/O devices, and (b) in automating data management to support computationally- demanding, data-intensive scientific applications. She has received an ERC Consolidator Award (2013), a Finmeccanica endowed chair from the Computer Science Department at Carnegie Mellon (2007), a European Young Investigator Award from the European Science Foundation (2007), an Alfred P. Sloan Research Fellowship (2005), an NSF CAREER award (2002), and nine best-paper awards in database, storage, and computer architecture conferences,. She holds a Ph.D. in Computer Science from the University of Wisconsin-Madison in 2000. She is an ACM fellow, an IEEE fellow, the Laureate for the 2018 Nemitsas Prize in Computer Science, and an elected member of the Swiss National Research Council. She has served as a CRA-W mentor, and is a member of the Expert Network of the World Economic Forum.

Querying Graph Databases with the GSQL Query Language

Alin Deutsch

Dept Computer Science & Engineering – University of California, San Diego (UCSD)
9500 Gilman Drive – 92093-0404 – La Jolla – CA – USA

deutsch@cs.ucsd.edu

Abstract. *This talk presents GSQL, a recent addition to the spectrum of query languages for expressing graph analytics. GSQL is a high-level yet still Turing-complete language whose syntax is inspired by SQL in order to reduce the learning curve for SQL programmers, while simultaneously supporting a Map-Reduce interpretation that is preferred by NoSQL developers and that is conducive to massively parallel evaluation. The talk will also provide some context on the graph query language landscape represented in modern systems.*

Alin Deutsch is a professor of Computer Science and Engineering at UC San Diego. His research is motivated by the data management challenges raised by database-powered applications. Alin's interests include query language design and optimization for various data models ranging from text to the relational and post-relational models (with particular emphasis on graph data). He also works on cross-model data integration and on automatic verification of business processes. Alin earned his PhD in Computer Science from the University of Pennsylvania, an MSc degree from the Technical University of Darmstadt (Germany) and a BSc degree from the Polytechnic University Bucharest (Romania). He is the recipient of the 2018 ACM PODS Test of Time Award, a Jean D'Alembert Fellowship from the University Paris-Saclay, the Alfred P.Sloan Fellowship, the ACM SIGMOD 2006 Top-3 Best Paper Award, and an NSF CAREER award.

Reducing Errors by Refusing to Guess (Occasionally)

Dennis Shasha

Department of Computer Science– New York University (NYU)
251 Mercer Street – 10012 – New York – NY – USA

shasha@cs.nyu.edu

Abstract. *We propose a meta-algorithm to reduce the error rate of state-of-the-art machine learning algorithms by refusing to make predictions in certain cases even when the underlying algorithms suggest predictions. Intuitively, our SafePredict approach estimates the likelihood that a prediction will be in error and when that likelihood is high, the approach refuses to go along with that prediction. Unlike other approaches, we can probabilistically guarantee an error rate on predictions we do make (denoted the decisive predictions). Empirically on seven diverse data sets from genomics, ecology, image-recognition, and gaming, our method can probabilistically guarantee to reduce the error rate to 1/4 of what it is in the state-of-the-art machine learning algorithm at a cost of between 11% and 58% refusals. Competing state-of-the-art methods refuse at roughly twice the rate of ours (sometimes refusing all suggested predictions).*

Dennis Shasha is a Julius Silver Professor of computer science at the Courant Institute of New York University (NYU) and an Associate Director of NYU Wireless. He works on meta-algorithms for machine learning to achieve guaranteed correctness rates, with biologists on pattern discovery for network inference; with computational chemists on algorithms for protein design; with physicists and financial people on algorithms for time series; on clocked computation for DNA computing; and on computational reproducibility. Other areas of interest include database tuning as well as tree and graph matching. Because he likes to type, he has written six books of puzzles about a mathematical detective named Dr. Ecco, a biography about great computer scientists, and a book about the future of computing. He has also written five technical books about database tuning, biological pattern recognition, time series, DNA computing, resampling statistics, and causal inference in molecular networks. He has co-authored over eighty journal papers, seventy conference papers, and twenty-five patents. He has written the puzzle column for various publications including Scientific American, Dr. Dobb's Journal, and the Communications of the ACM. He is a fellow of the ACM and an INRIA International Chair.

Author Index