Lathe: Light-Weight Keyword Query Processing over Multiple Databases

Pericles de Oliveira[1], Altigran da Silva[1], Edleno Moura[1], Gilberto Santos[1]

[1] IComp/UFAM

{pericles.oliveira}@nokia.com, alti,edleno,gilberto@icomp.ufam.edu.br

Abstract. *We present a new R-KwS system called* Lathe*, which, is able to efficiently deal with such conditions. Lathe is based on a probabilistic model and principled strategies which allow pruning unlike candidate solutions for a keyword query very early in the process. As a result, the system is able to produce accurate answers very quickly in comparison to current systems. Lathe is able to handle many distinct databases by trying a same query over multiple different target databases at once, and providing answers for all those databases that have an answer to the query.*

1. Introduction

Among the most well-known existing R-KwS systems, there are those that take advantage of the basic functionality of the underlying RDBMS, and produce, from the keywords supplied in the queries, relational join expressions that try to fulfill the user's information needs. These relational expressions, called *Candidate Networks of Relations*, or simply *Candidate Networks* (*CNs*) [Hristidis et. al., 2002], are coded as SQL queries which, when executed in a target RDBMs, are expected to produce answers relevant to the keyword query posed by the user. Formally, CNs are joining trees of relation derived from a graph representing the schema of the target database. Thus, systems that rely on them are usually called *Schema Graph* R-KwS systems. Examples of these systems are DISCOVER [Hristidis et. al., 2002], CNRank [de Oliveira et al., 2015] and KwS-F [Baid et. al., 2010] [1].

We present a new Schema Graph R-KwS system called *Lathe*[2][3], which is able to efficiently deal with such conditions. Lathe is based on a probabilistic model and principled strategies, which allow pruning unlike Candidate Networks for a keyword query very early in the process. As a result, the system is able to produce accurate answers very quickly in comparison to current systems. Thus, our system represents a viable alternative to overcome a critical issue for the wide adoption of R-KwS systems in real applications

Lathe is able to seamlessly handle multiple target databases. Given a query, it is able to try a same input query over many databases at once, and providing answers for all those databases that are likely to have relevant results to the query. This not only simplifies the process of publishing new databases on-line, but also empowers the user to explore different databases from which she barely known the domain.

[1] There are in fact many more important systems that we did not cite only for saving space.

[2] This name alludes to the fact that the system automatically assigns an structure to an unstructured input keyword query, that is, it "shapes" the query.

[3] Funding by projects TTDSW (FAPEAM/CNPq), e-Vox Pesquisa (FAPEAM), e-Spot (CNPql), and by individual CNPq grants.

In the rest of this paper, we first present an overview on Lathe's architecture in Section 2. Finally, Section 3 describes how the demo session will be carried out.

2. System Overview

Lathes' architecture is illustrated in Figure 1. Given an input query, Lathe, as all other

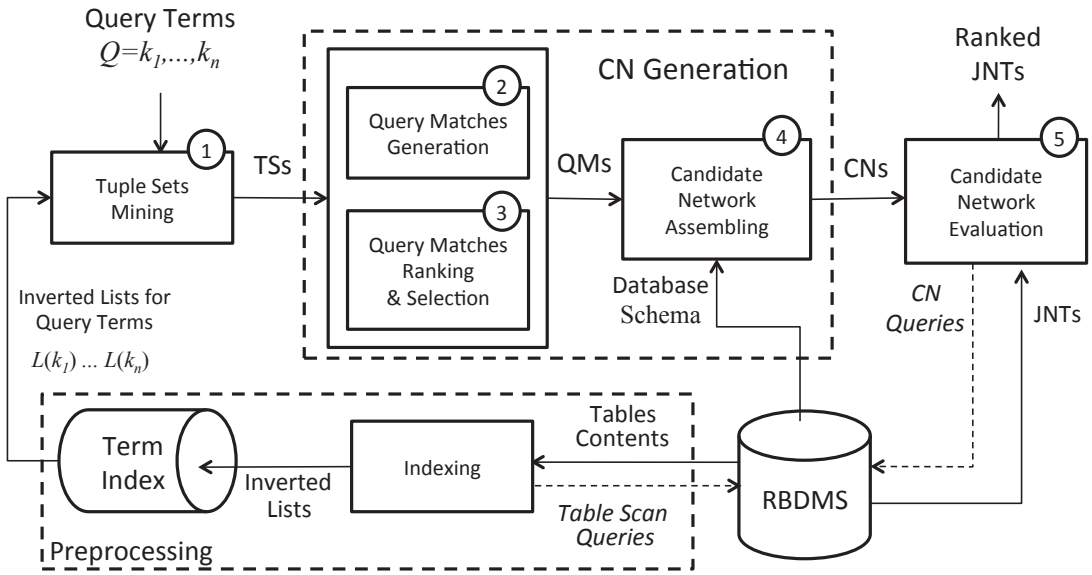

Figure 1. Lathe: architecture and main functioning.

Schema Graph R-KwS systems, first generates a number of Candidate Networks and then evaluates these CNs to produce answers. The main diference between Lathe and all previous systems lies in how it generates CNs. So far, all existing systems in the literature use for this task the CNGen algorithm proposed in [Hristidis et. al., 2002] for the DISCOVER system. In some cases, there is an additional step to somehow improve the set of CNs generated, as it occurs in KwS-F [Baid et. al., 2010] and CNRank [de Oliveira et al., 2015]. Lathe adopts a new approach for generating CNs, as we detail below.

Intuitively, a CN is a relational join expression that "connects" subsets of relations from the database whose tuples contain one or more keywords of the query. The "connections" are derived from referential integrity, i.e., PK/FK, constraints, which may involve additional relations. By having a DMBS to evaluate CNs, we obtain semantically meaningful answers as joined tuples which contain the query keywords.

As an example, consider the query "*denzel washington gangster*" and suppose we want to execute it over a relational database containing data on movies from the well-known Internet Movie Databases (IMDb). A possible CN for this query is given by:

$$\sigma_{\text{name} \supseteq \{gangster\}} \text{CHAR} \bowtie_{\text{id=cid}} \text{CAST} \bowtie_{\text{cid=pid}} \sigma_{\text{name} \supseteq \{denzel,washington\}} \text{PER} \quad (1)$$

where CHAR stores names of movie characters, PER stores data on persons (i.e., actors, actresses, directors, etc.) and CAST associates persons to the characters they play on movies. The join conditions in this expression are derived from PK/FK constraints.

Following the terminology introduced in [Hristidis et. al., 2002], the operands of the join operations in a CN are called *Tuple Sets*. Operands whose tuples contain the keywords specified in the query, such as those defined by selection operations over CHAR and PER in Equation 1, are called *Non-free Tuple Sets*. The remaining operands, such as CAST in Equation 1, are called *Free Tuple Sets*, since they not contain any of the keywords. For simplicity, in this paper we use *Tuple Sets* to refer to *Non-free Tuple Sets*, while *Free Tuple Sets* are referred explicitly.

The complexity of the CN generation problem is mainly due to two factors: (1) There can be multiple tuple sets for each subset of terms of the query. As a consequence, there may be a large number of ways of combining these tuple sets so that all terms of the query are covered; (2) Given a set of tuple sets that cover the terms of the query, there can be many distinct ways of connecting them through PK/FK constraints and free tuple sets.

3. Demonstration Plan

For the demo session, we prepared experiments using databases previously used in evaluations of R-KwS systems to highlight Lathe's main features. The experiments consist of tasks in which a user submits a keyword query, and, for this query, the system generates a number of Candidate Networks (CNs) and then evaluates these CNs to produce a ranking of Joint Networks of Tuples (JNTs) as a final answer. The goal is to demonstrate the behavior of the system in two aspects: the quality of the CNs produced, along with their impact on the quality of the results obtained when these CNs are evaluated; and the system's performance and scalability.

In each task, the user enters a keyword query and receives the ranked JNT provided by the system, with an indication of the time elapsed. The system also allows the user to examine details of each step of the keyword query processing, as described in Section 2 (see Figure 1). This includes the presentation of intermediary results, with summaries that demonstrate the behavior of the systems in terms of effectiveness and performance. The steps and the results that can be examined for each query execution are described in Table 1.

Table 1. Steps to be explored in the demo session.

Step	Results Presented
TS Mining	A list of the TSs are mining from termsets; a summary with the number of TSs generated and the time elapsed.
QM Generation	A list of QMs generated; a summary with the number QMs generated and the time elapsed.
QM Ranking	A ranking of the QMs generated; a summary with values for metrics such as precision, recall, MAP and RR, and the time elapsed.
CN Assembling	A list of the CNs generated from each QM; a summary with the number of CNs generated and the time elapsed.
CN Evaluation	The issuing of each CN to be evaluated on the DMBS; a summary with the number of JNTs obtained and the time elapsed.
JNT Ranking	The top-10 JNT; a summary with values for metrics such as precision, recall, MAP and RR, and the time elapsed.

We use 4 datasets: IMDb, Mondial, Wikipedia and DBLP, which were previously used for the experiments used in [Coffman et. al., 2010], [Luo et. al., 2007], [de Oliveira et al., 2015] [de Oliveira et al., 2018] and many other previous work. In Table 2 we present some details on these datasets, including their size (in Megabytes), the number of relations, the total number of tables and the number of Referential Integrity Constrains (RIC) in their schemas.

Table 2. Datasets used in the demo session.

Dataset	Size (MB)	Relations	Tuples	RIC
Mondial	9	28	17,115	104
IMDb	516	6	1,673,074	4
Wikipedia	550	6	206,318	5
DBLP	40	6	878,065	6

Queries used in the tasks were selected from 3 query sets from different sources: (1) Coffman: Queries used in the experiments reported in [Coffman et. al., 2010], targeted to datasets IMDb, Mondial and Wikipedia; (2) SPARK: Queries used in the experiments reported in [Luo et. al., 2007], targeted to datasets IMDb, DBLP and Mondial; (3) INEX: Queries originally specified for the INEX 2011 challenge[4], targeted to datasets IMDb which were adapted for searching in relational databases.

For didactic purposes, a companion poster showing the steps being executed is used to illustrate the running of the tasks. Attendees will also be invited to propose open, i.e., non-predefined tasks. Furthermore, besides trying the system during the session, interested attendees will also be able to experience the on-line system themselves on their on devices.

The demo session will be conducted on-line through a browser interface. The system is hosted on the cloud, in a regular machine with not parallel processing, running Centos 6.3 Linux. We use PostgreSQL as the underlying RDBMS with a default configuration. All implementations use the Java language.

Acknowledgments Work partially funded by projects eSpot (CNPq 461231/2014-0), SocSens (CAPES/PCGI 88887.130299/2017-01), CARECO (PROCAD/CAPES 88881.068507/2014-01), ATMOSPHERE (EC/H2020 grant no. 777154 & RNP/MCTIC acordo 51119), and by authors' individual grants from CNPq.

References

Baid et. al. (2010). Toward scalable keyword search over relational data. *PVLDB.*, 3(1-2):140–149.

Coffman et. al. (2010). A framework for evaluating database keyword search strategies. In *CIKM*, pages 729–738.

de Oliveira, P., da Silva, A. S., and de Moura, E. S. (2015). Ranking candidate networks of relations to improve keyword search over relational databases. In *ICDE*, pages 399–410.

de Oliveira, P., da Silva, A. S., de Moura, E. S., and Rodrigues, R. (2018). Match-based candidate network generation for keyword queries over relational databases. In *ICDE*, pages 16–19.

Hristidis et. al. (2002). DISCOVER: keyword search in relational databases. In *VLDB*, pages 670–681.

Luo et. al. (2007). SPARK: Top-k keyword query in relational databases. In *SIGMOD*, pages 115–126.

[4] `https://inex.mmci.uni-saarland.de`