Keyword Search over Data Service Integration for Accurate Results

Vidmantas Zemleris[1], Valentin Y Kuznetsov[2] and Peter Kreuzer[3]

[1]Faculty of Mathematics and Informatics, Vilnius University, Lithuania and CMS Experiment at CERN, Geneva, Switzerland
[2]Cornell University, USA
[3]Rheinisch-Westfaelische Tech. Hoch., Germany

E-mail: [1]`vidmantas.zemleris@cern.ch`

Abstract. The goal of the virtual data service integration is to provide a coherent interface for querying a number of heterogenous data sources (e.g., web services, web forms, proprietary systems, etc.) in cases where accurate results are necessary. This work explores various aspects of its usability.

Querying is usually carried out through a structured query language, such as SQL, which forces the users to learn the language and to get acquainted with data organization (i.e. the schema) thus negatively impacting the system's usability. Limited access to data instances as well as users' concern with accurate results of arbitrary queries present additional challenges to traditional approaches (such as query forms, information retrieval, keyword search over relational databases) making them not applicable.

This paper presents a keyword search system which deals with the above discussed problem by operating on available information: the metadata, such as the constraints on allowed values, analysis of user queries, and certain portions of data. Given a keyword query, it proposes a ranked list of structured queries along with the explanations of their meanings. Unlike previous implementations, the system is freely available and makes no assumptions about the input query, while maintaining its ability to leverage the query's structural patterns - in case they exist. The system is discussed in the context of CMS data discovery service where the simplicity and capabilities of the search interface play a crucial role in the ability of its users to satisfy their information needs.

1. Introduction

In *Virtual Data Integration* (EII), data physically stays at its origin, and is requested only on demand, usually, through structured query languages such as *SQL*. EII performs a number of transformations on queries and their results (eliminating the inconsistencies in data formats, naming; combining the results, etc.) which allows querying the sources in a coherent way. **However, this** requires its users to learn the query language and to get acquainted with data organization (i.e. the mediated schema) thus negatively impacting the system's usability.

The objective of this work is to investigate the keyword search proposing a ranked list of queries, as a more intuitive alternative, which, in fact, received relatively little attention in the field of data **(service)** integration[1].

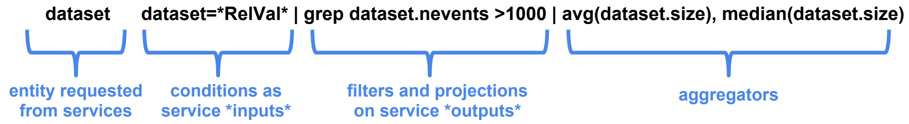
Virtual integration presents an additional challenge, since only limited access to the data instances is available, rendering the traditional methods that return data-tuples inapplicable (e.g. Information Retrieval or Keyword search over relational databases).

Building on the experience gained while working on an EII system at the *CMS Experiment* at *CERN*, we will focus on the implementation of keyword search, also touching and the mechanisms for user feedback and some more distant topics such as usability and performance of an EII.

1.1. The Data Aggregation System

The *"CMS Data Aggregation System" (DAS) [2, 3]* allows integrated access to a number of proprietary data-sources through simple structured queries eliminating the inconsistencies in entity naming, data formats and combining the results. It uses the *Boolean Retrieval Model* as users are often interested in retrieving ALL the items matching their query.

The queries are formed specifying the entity the user is interested in (e.g. dataset, file, etc) and providing selection criteria (e.g. attribute=value, attribute *between* [v1, v2]). The results could be later 'piped' for further filtering, sorting or aggregation (min, max, avg, sum, count, median), e.g.:



```
dataset dataset=*RelVal* | grep dataset.nevents >1000 | avg(dataset.size), median(dataset.size)
```

entity requested from services
conditions as service *inputs*
filters and projections on service *outputs*
aggregators

As seen above, DASQL closely corresponds to the physical execution flow: based on the requested entity and the conditions on service inputs, DAS decides the set of services to be queried[1]. Then, after retrieving, processing and merging the results from services, the filters and projections, and aggregators are applied. The results are cached for subsequent uses.

2. From Keywords to Queries

2.1. Tokenization

2.2. Scoring individual keywords (entry points)

2.3. Ranking function

3. Generating the results

3.1. via Exhaustive search

allows filtering only those that DIS supports.

3.2. via Solving weighted-bipartite assignments

3.3. Combination(?)

exhaustive for part that is well

4. Our implementation

cython

5. Future work

6. Related work

7. Conclusions

References

[1] Guerrisi V, La Torre P and Quarteroni S 2012 *Search Computing* 82–97

[2] Kuznetsov V, Evans D and Metson S 2010 *Procedia Computer Science* **1** 1535 – 1543 ISSN 1877-0509 iCCS 2010 URL `http://www.sciencedirect.com/science/article/pii/S1877050910001730`

[3] Ball G, Kuznetsov V, Evans D and Metson S 2011 *Journal of Physics: Conference Series* **331** 042029 URL `http://stacks.iop.org/1742-6596/331/i=4/a=042029`

[1] including pre-defined "virtual services", which feed results from one service into inputs of the others