Distributed Semantic Query Tool and Query PreProcessor

Janne Saarela, Pasi Tiitinen (Profium)

Philipp Obermeier, Sebastian Dill (FUB)

15/05/09

Outline

- Year 2 and reviewers' comments
- Work achieved in year 3
  - Distributed Semantic Query Tool
  - Query PreProcessor

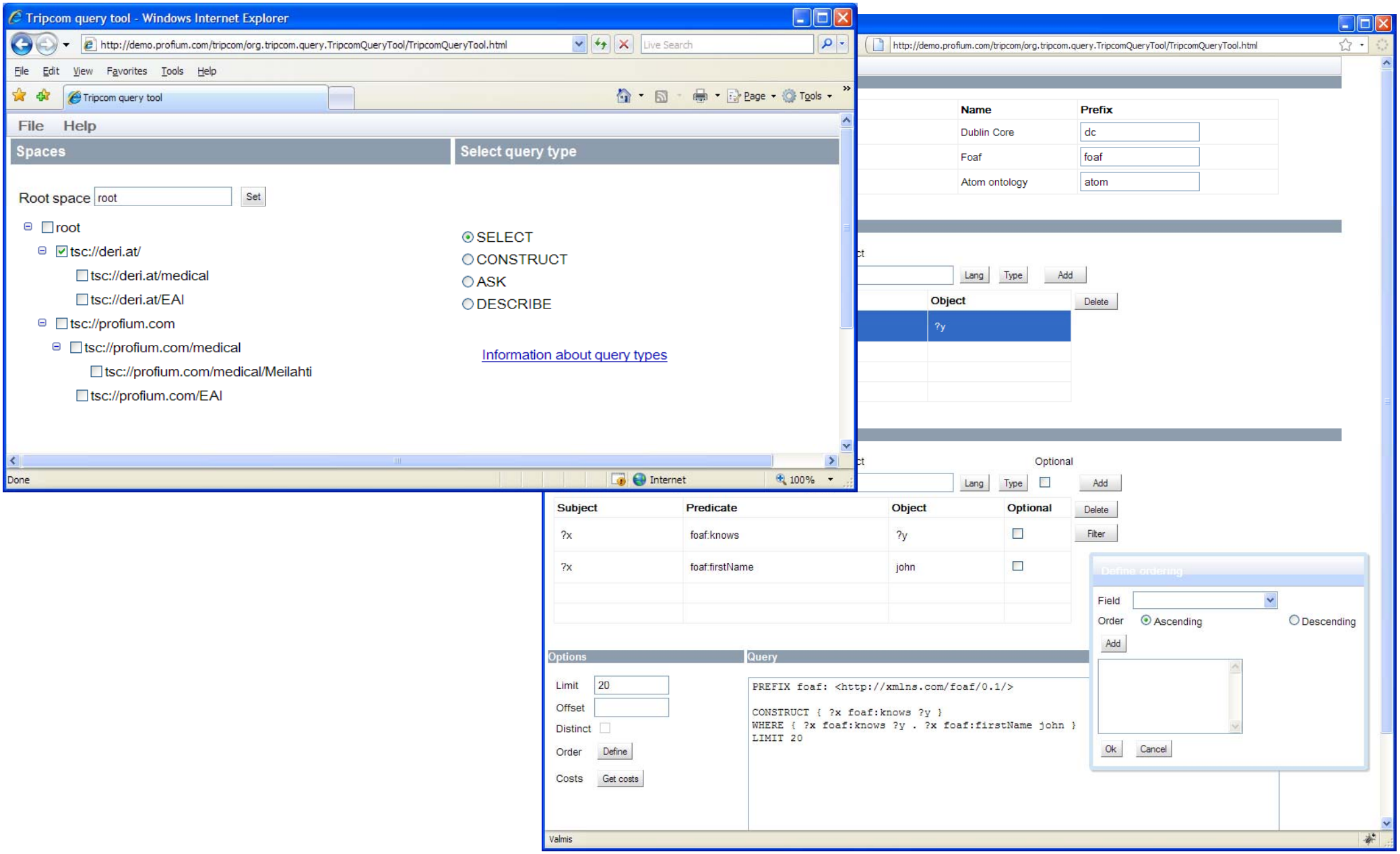
Year 2 and comments

- In year 3 we developed software to
  1. assist developers to make use of Triple Space technology more easily and
  2. optimize query evaluation in a distributed environment
- Following issue was raised at the year 2 review:
  - *While the appropriateness of the [query evaluation] cost model still has to be shown, a well designed cost model will be of value not only in the context of the Tripcom project, but can also provide benefits to other developers of distributed RDF infrastructures.*
  - *-> the cost model has been implemented in the Query PreProcessor*

Distributed Semantic Query Tool (DSQT)

- Requirements for the tool:
  - must support activities of the Triple Space developers by allowing its users to write valid SPARQL queries.
  - must allow its users to evaluate SPARQL queries against one or many triple space(s).
  - must allow its users to visualize query results in an environment where the query expression can be further edited and evaluated.
  - the constructed query expression and resulting query results must both be storable in standard format in local hard drive for later re-use.
  - the query execution cost may be explained to the user prior to its execution.
  - should show metadata about the Triple Spaces being queried.

Distributed Semantic Query Tool (DSQT)



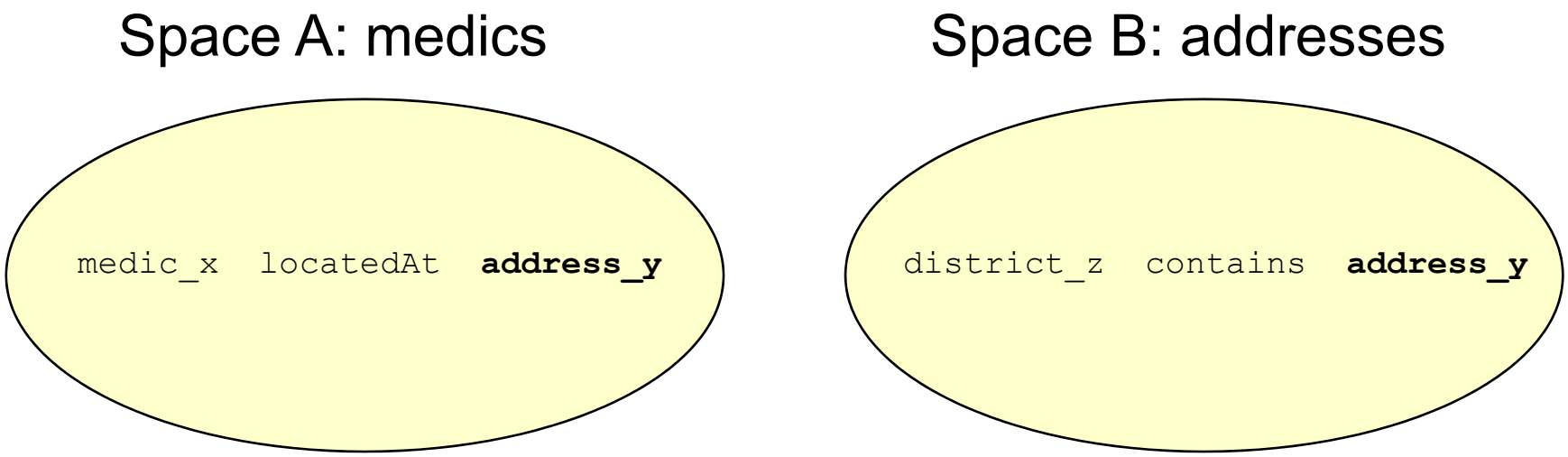
DSQT

- Thin-client application developed using Google Web Toolkit (GWT)
- A wizard approach for creating a SPARQL query which can be ’explained’ and ’executed’ much in the same way as in SQL based environments.
- Triple Space specific features allow the user to understand the space structure against which the query is executed.

- DSQT makes use of Query PreProcessor…

QPP- Motivation (I)

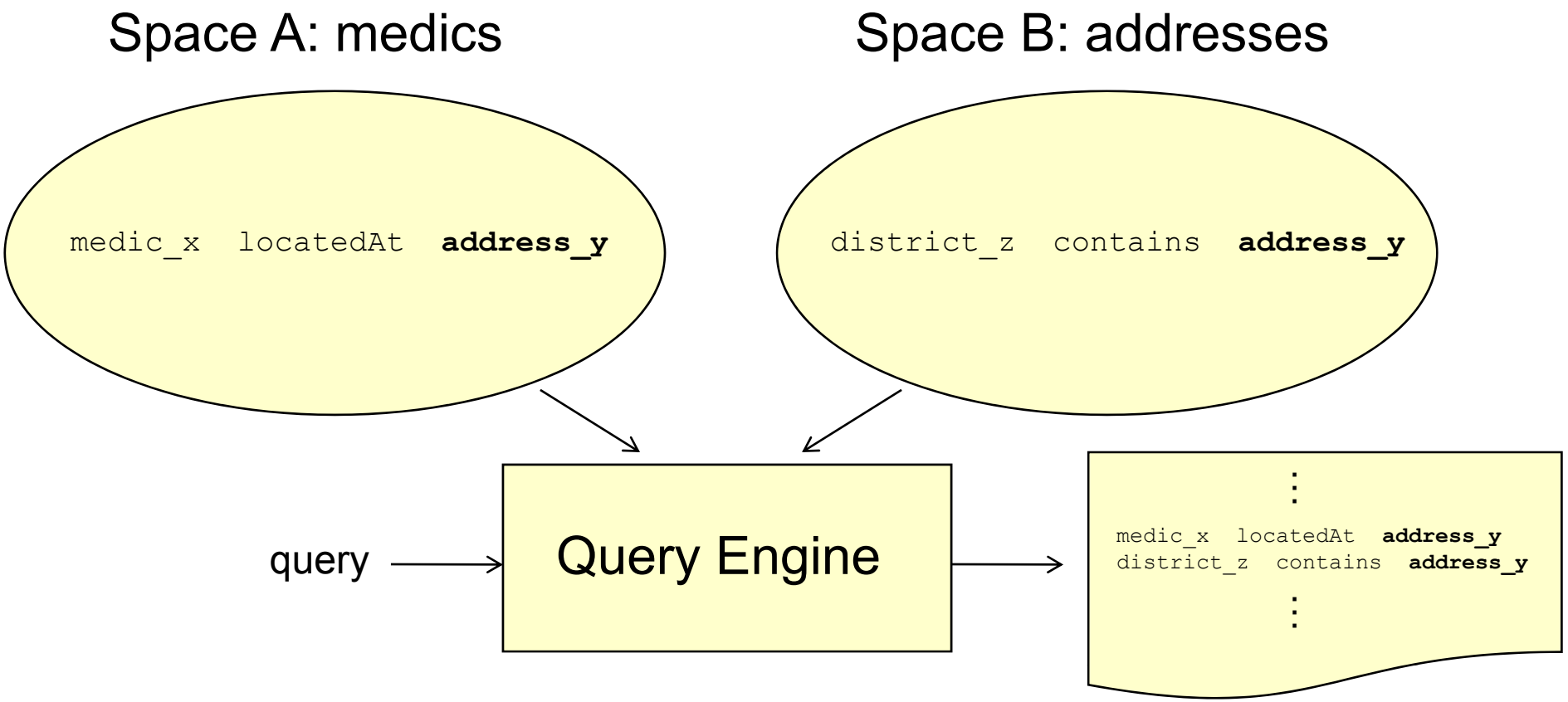
- A Triple Space System comprises several machines (kernels) hosting RDF data sets (triplespaces) with SPARQL endpoint (TS API, TS Adapter, SPARQL CONSTRUCT queries as rd templates)
- Related data is often located in different spaces hosted on separated kernels



- However: ‘rd without space’ is evaluated against one space at most

QPP - Motivation (II)

Engine for federated querying upon the integrated RDF data set of multiple RDF data sets with SPARQL endpoint (here spaces)



QPP – Concept (I)

Query Preprocessor (QPP), a federated query engine for SPARQL:

- provides the means to distribute the evaluation of a query across multiple RDF data sets with SPARQL endpoint, optimized for execution time
- splits query into sub-queries and sends them to SPARQL endpoints (detected by DHT overlay) for execution.

QPP – Concept (II)

Significant aspects on query planning, optimization and execution:

- Data sets indexed by DHT overlay adopted from Distribution Manager
- Cost estimation based on RDF stats for each data set
- Intra-query adaptiveness, i.e., query planning and execution interplay
  - Planning part: partitioning of queries (BGPs) into sub-queries controlled by connectivity and existence of adequate data sets (i.e., the ones, which can resolve the searched triple patterns while yielding acceptable costs)
  - Execution part: shipping of selected sub-queries to selected data sets for evaluation, return of results to QPP
- Trade-off of perfect recall against performance: Choice of single data set with median amount of solutions instead of all adequate data sets; Triple Space System allows non-perfect recall

QPP - Evaluation (I)

eHealth evaluation scenario:

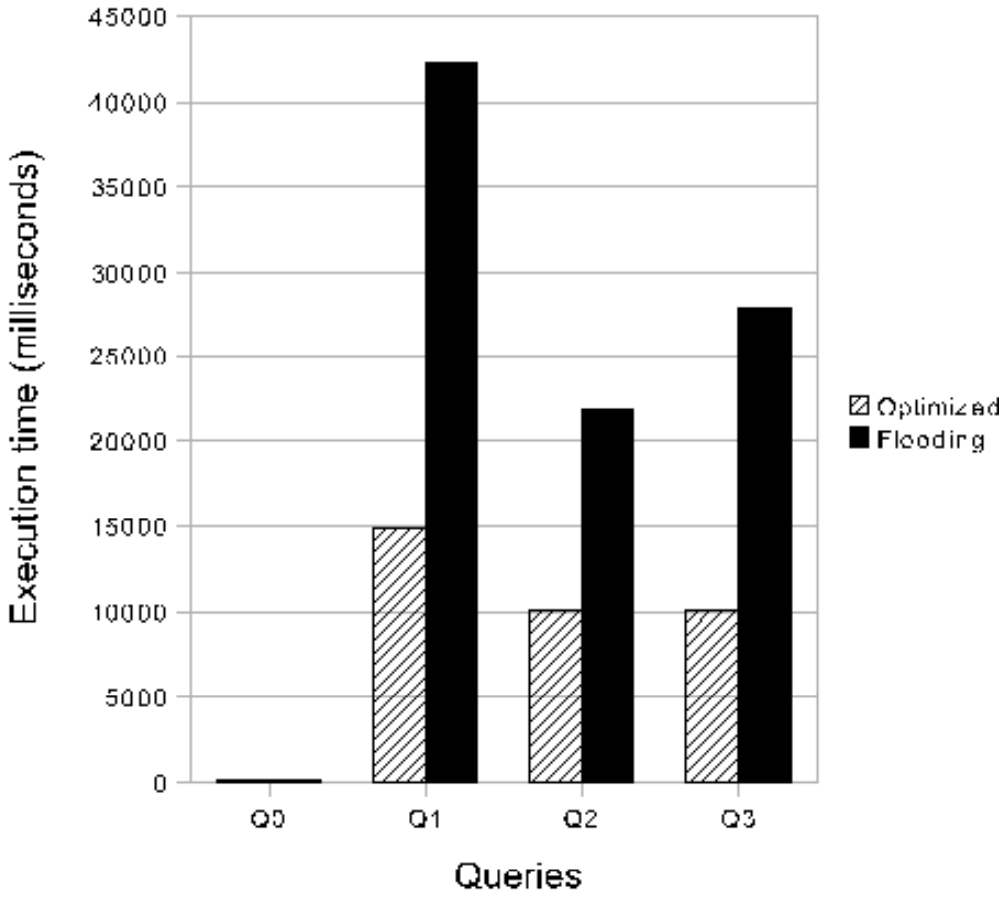
- Global data set with 13.5 million triples split up into 12 data sets: 3 exclusive data sets for each concept (medics, treatments, drugs, districts) were distributed to 4 machines
- These 4 machines (1 CPU Dual Core, 2GB Ram, Debian Linux) each hosted 3 data sets with a SPARQL endpoint (i.e. 3 instances of Open Link Virtuoso on each machine)
- e.g. test query 1

```
SELECT ?med ?treatment
WHERE { ?med medic:locatedAt ?address .
district:district_1 district:contains ?address .
?med medic:provides ?treatment . }
```

where medic and district objects are only retrievable from different endpoints

QPP – Evaluation Results

- Queries can now be evaluated against the integration of multiple data sets
- Tremendous performance gain in contrast to a naive distributed querying approach (i.e. asking all adequate data sets)
- Performance gain is available outside Triple Space environment for existential queries.



Conclusion and Future Works

QPP

- provides the means for federated adaptive query processing across SPARQL endpoints and, thus, across triplespaces
- highly optimized for distributed querying scenarios in which only a small fraction, i.e., a few solution mappings, of the perfect recall is sufficient; acts much more efficiently than naive approache for querying multiple data sources/spaces

Future research on

- further advanced methods for federated querying of SPARQL endpoints