PERFORMANCE ANALYSIS AND OPTIMIZATION OF SEMANTIC QUERIES

Philipp Hertweck, Erik Kristiansen, Tobias Hellmund, Jürgen Moßgraber

Nice, 2020

Contact

Philipp Hertweck

E-mail: philipp.hertweck@iosb.fraunhofer.de

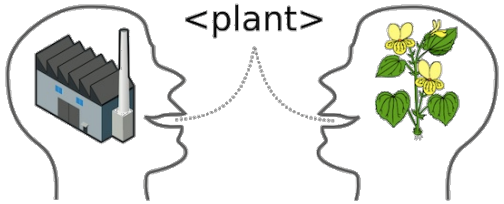
Phone: (+49) 721 6091-372

About me

- M. Sc. in Computer Science from Karlsruhe Institute of Technology (KIT), Germany
- Research associate: Fraunhofer Institute of Optronics, System Technologies and Image Exploitation (IOSB), Karlsruhe

- Research topics:
  - Design & Implementation of distributed systems
  - Semantic modeling & -data integration
  - Internet of Things use cases

Introduction



source: https://www.peterkrantz.com/2010/semantic-interoperability/

- Semantic data integration
  - Combination of different data sources
  - Unified understanding

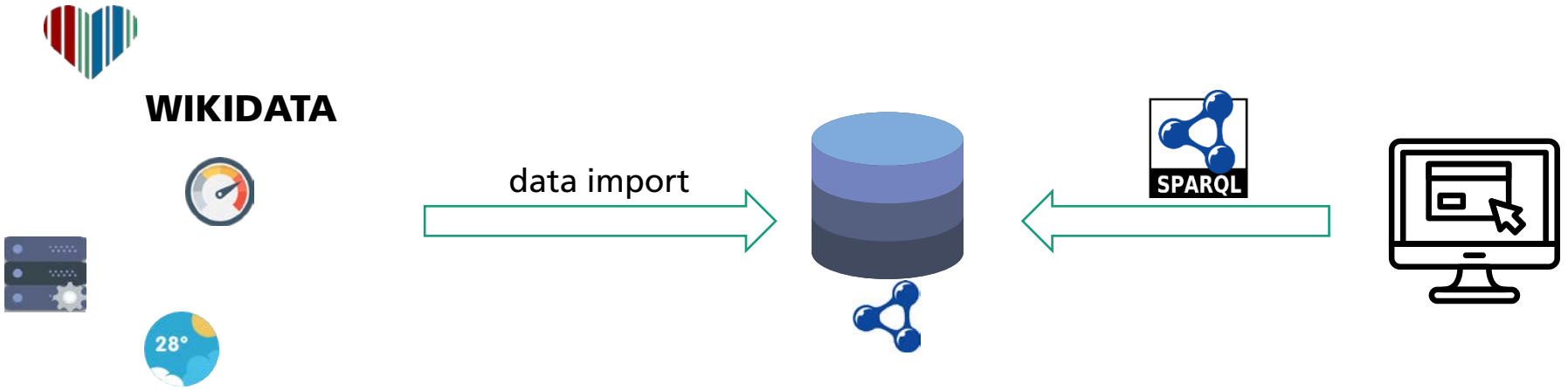
- Implemented in various domains
  - Cultural Heritage
  - Smart City
  - Disaster management
  - …

source: https://beaware-project.eu

source: http://www.heracles-project.eu

source: http://www.heracles-project.eu

- Semantic data integration



- Challenge
  - „live“-data
  - volatile data

→ Performance is a key factor

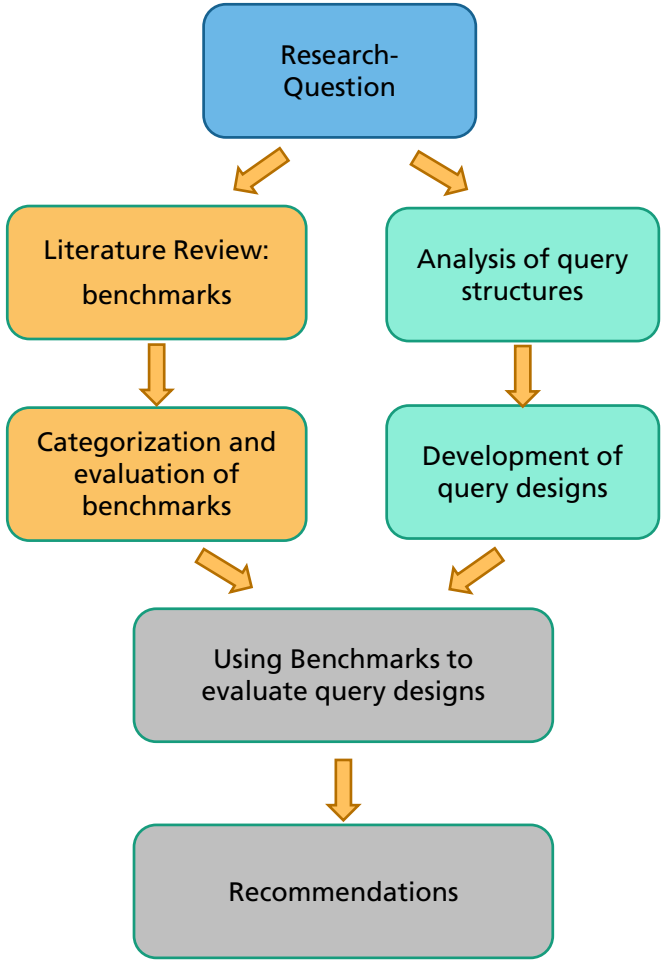
- Developers not aware of SPARQL performance characteristics

images: Flaticon.com, W3C

Related work

- Rietveld et al. showed 72,66% of analyzed queries are formulated inefficiently
- Performance optimization well known from relational data bases
- Yet, no easily applicable rules for SPARQL available

Our approach



SPARQL-Benchmarks: Evaluation criteria

- User defined ontology
- **Data generator**
- Query generator
- **User defined queries**
- **Query execution**
- **Code availability**
- Last update
- **License**

SPARQL-Benchmarks: Result

Name	User ontology	Data generator	User queries	Query generator	Executor	Code available	Last Update	License
Berlin SPARQL Benchmark BSBM [14]	No	Yes	Yes	No	Yes	✓	2012	Apache 2.0
Lehigh University Benchmark LUBM [15]	No	Yes	Yes	No	Yes	✓	2004	GPL 2.0
FedBench [16]	No	No	No	No	Yes	✓	2013	LGPL
Feasible [17]	No	No	Yes	Yes	No	✓	2018	AGPL
LargeRDFBench [18]	No	No	Yes	No	No	✓	2018	AGPL
University Ontology Benchmark UOBM [19]	No	Yes	No	No	No	✗	2005	GPL 2.0
SPARQL Performance Benchmark [20]	No	Yes	No	No	No	✓	2009	Berkeley License
Social Network Intelligence Benchmark [21]	No	Yes	No	No	Yes	✗	2015	GPL 3.0
Linked Data Integration Benchmark [22]	No	Yes	Yes	No	No	✓	2012	BSD
Linked Open Data Quality Assessment [23]	No	No	No	No	Yes	✓	2012	BSD
LinkBench [24]	Yes	Yes	Yes	No	Yes	✓	2015	Apache 2.0
Waterloo SPARQL Diversity Test Suite [25]	No	Yes	Yes	Yes	No	✓	2014	MIT
Semantic Publishing Benchmark [26]	No	Yes	No	No	Yes	✓	2019	Apache 2.0
IGUANA [27]	Yes	No	Yes	No	Yes	✓	2019	AGPL

Identifying Query Patterns

```
select ?review ?rating2 where {
    ?review bsbm:rating1 ?rating1.
    ?review bsbm:rating2 ?rating2
    filter (?rating1 >= %rating1% &&
            ?rating2 < %rating2%)
}
```

```
select ?review ?rating2 where {
    ?review bsbm:rating1 ?rating1.
    filter (?rating1 >= %rating1%)
    ?review bsbm:rating2 ?rating2.
    filter (?rating2 < %rating2%)
}
```

Filter size

Query patterns I

- Number of results
  - # triples in triple store
- Limiting results
  - `LIMIT`
- Projection
  - `SELECT *` and `SELECT ?var`
- String functions
  - `RegEx` and `STRSTARTS`
- Filter size
- Filter position
- String filter
  - Numerical or text
- Inverse
  - `?r rev:r ?p` and `?p ^rev:r ?r`
- Variable types
  - Adding `rdf:type`
- Optional
  - `OPTIONAL` and `UNION`

Query patterns II

- Graph structure
  - Structure and attributes
- Triple order
- Limit in subselect
- Distinct
  - `DISTINCT` and `REDUCES`
- Minus
  - `filter(not exists {…})` and `MINUS`
- Path
  - relation and `^bsbm:reviewFor/rev:reviewer`

Evaluation setup

- Triple Store: Apache Fuseki 
- Commodity hardware
- Generated datasets
  - 4 million triples
    - 7 GB, 2.45 GB heap size
  - 1.8 million triples
    - 7 GB, 4 GB, 1.11 GB heap size

Results

Query Pattern	4 million		1.8 million			
	7 GB	2.45 GB	7 GB	4 GB	1.11 GB	VI
Number of results	✓ 2	✓ 2	✓ 2	✓ 2	✓ 2	1
Limiting result	✓ 2	✓ 2	✓ 2	✓ 2	✓ 2	1
Projection	✓ 3	✓ 2	✓ 3	✓ 3	✓ 3	2
String functions	✓ 2	✓ 2	✓ 2	✓ 2	✓ 2	6
Filter size	✗	✗	✗	✗	✗	-
FilterPos	✗	✓ 2	✗	✗	✗	-
String filter	✓ 1	✓ 2	✓ 1	✓ 1	✓ 2	5
Inverse	✗	✗	✗	✓ 1	✗	-
Variables type	✓ 1	✓ 1	✓ 1	✓ 1	✓ 1	4
Optional	✗	✗	✓ 1	✓ 1	✗	-
Graph structure	✓ 2	✓ 2	✗	✗	✗	-
Triple order	✓ 2	✓ 2	✓ 2	✓ 2	✓ 2	3
Limit in subselect	✓ 1	✓ 1	✗	✗	✗	-
Distinct	✗	✓ 2	✓ 2	✓ 2	✓ 2	7
Minus	✓ 2	✓ 2	✓ 2	✓ 2	✓ 2	8
Paths	✗	✗	✗	✗	✓ 2	-

- Same internal representation
  - Filter Size
  - Filter Position
  - Inverse
- No change
  - Triple order

Results

Query Pattern	4 million		1.8 million			
	7 GB	2.45 GB	7 GB	4 GB	1.11 GB	VI
Number of results	✓ 2	✓ 2	✓ 2	✓ 2	✓ 2	1
Limiting result	✓ 2	✓ 2	✓ 2	✓ 2	✓ 2	1
Projection	✓ 3	✓ 2	✓ 3	✓ 3	✓ 3	2
String functions	✓ 2	✓ 2	✓ 2	✓ 2	✓ 2	6
Filter size	✗	✗	✗	✗	✗	-
FilterPos	✗	✓ 2	✗	✗	✗	-
String filter	✓ 1	✓ 2	✓ 1	✓ 1	✓ 2	5
Inverse	✗	✗	✗	✓ 1	✗	-
Variables type	✓ 1	✓ 1	✓ 1	✓ 1	✓ 1	4
Optional	✗	✗	✓ 1	✓ 1	✗	-
Graph structure	✓ 2	✓ 2	✗	✗	✗	-
Triple order	✓ 2	✓ 2	✓ 2	✓ 2	✓ 2	3
Limit in subselect	✓ 1	✓ 1	✗	✗	✗	-
Distinct	✗	✓ 2	✓ 2	✓ 2	✓ 2	7
Minus	✓ 2	✓ 2	✓ 2	✓ 2	✓ 2	8
Paths	✗	✗	✗	✗	✓ 2	-

1. Small result set
2. Use projections
3. Reduce intermediate results
4. Don not add additional types
5. Avoid filtering for text
6. Use STR-functions
7. Use REDUCE instead of DISTINCT
8. Use Minus-Operator instead of Filter

Conclusion

- Review of SPARQL-Benchmarks
- 9 easy applicable recommendations

- Future Work
  - Extend to more triple stores
  - Analysis of other influencing factors, e.g. Caching
  - Analyze influence of ontology structure

QUESTIONS?

Philipp Hertweck

Fraunhofer IOSB, Karlsruhe

E-mail: philipp.hertweck@iosb.fraunhofer.de

Tel.: (+49) 721 6091-372