

2010-11 Activity Report

Bruno Paiva Lima da Silva

Supervisors: Jean-François BAGET & Madalina CROITORU

Université Montpellier 2

- 1 Research problem
- 2 Encodings & Translations
- 3 Preliminary results
- 4 Future work
- 5 Questions

Table of Contents

- 1 Research problem
- 2 Encodings & Translations
- 3 Preliminary results
- 4 Future work
- 5 Questions

Research problem

(1) Ontological conjunctive query answering

Decision problem

Research problem

(1) Ontological conjunctive query answering



Knowledge base

Decision problem

Research problem

(1) Ontological conjunctive query answering

Factual knowledge
(Very often a DB)

Knowledge base

Decision problem

Research problem

(1) Ontological conjunctive query answering

Factual knowledge
(Very often a DB)

Ontology
(Universal knowledge)

Knowledge base

Decision problem

Research problem

(1) Ontological conjunctive query answering

Factual knowledge
(Very often a DB)

Ontology
(Universal knowledge)

Query
(Conjunctive query)

Knowledge base

Decision problem

Research problem

(1) Ontological conjunctive query answering

Factual knowledge
(Very often a DB)

Ontology
(Universal knowledge)

Query
(Conjunctive query)

Knowledge base

Decision problem

(1) "Is there an answer to the query in the knowledge base"?

Research problem

(1) Ontological conjunctive query answering

Factual knowledge
(Very often a DB)

Ontology
(Universal knowledge)

Query
(Conjunctive query)

Knowledge base

(2) Logical form

Logical fact \mathcal{F}
(Conjunction of positive atoms)

Ontology \mathcal{O}
($\forall\exists$ -rules)

Query \mathcal{Q}
(Conjunctive query)

Decision problem

(1) "Is there an answer to the query in the knowledge base"?

Research problem

(1) Ontological conjunctive query answering

Factual knowledge
(Very often a DB)

Ontology
(Universal knowledge)

Query
(Conjunctive query)

Knowledge base

(2) Logical form

Logical fact \mathcal{F}
(Conjunction of positive atoms)

Ontology \mathcal{O}
($\forall\exists$ -rules)

Query \mathcal{Q}
(Conjunctive query)

Decision problem

(1) "Is there an answer to the query in the knowledge base"?

(2) $\{\mathcal{F}, \mathcal{O}\} \models \mathcal{Q}$?

Research problem

$$\mathcal{F} \models Q$$

... iff there is a substitution \mathcal{S} associating every term of the query to a term in the facts.

Problem: **Finding substitutions**
(Also known as **ENTAILMENT**)

Research problem

$$\mathcal{F} \models Q$$

... iff there is a substitution \mathcal{S} associating every term of the query to a term in the facts.

Problem: **Finding substitutions**
(Also known as **ENTAILMENT**)

$$\{\mathcal{F}, \mathcal{O}\} \models Q$$

... iff after being enriched by \mathcal{O} , there is a substitution \mathcal{S} associating every term of the query to a term in the facts.

Problem: **Applying rules, Finding substitutions**
(Also known as **RULE-ENTAILMENT**)

Equivalences

According to the set of rules \mathcal{O} we choose, we retrieve a semantical equivalence from our problem into others that are or have already been studied in the litterature.

- If \mathcal{O} is empty, our problem becomes equivalent to the **ENTAILMENT** problem in RDF language.
- If \mathcal{O} is a set of \forall -rules, we enter the RDFS, Datalog and Conceptual Graphs (CGs) scope.
- If \mathcal{O} is a set of $\forall\exists$ -rules, we obtain an equivalence to the problems found in Datalog \pm and CGs with rules.

Rules

A rule contains two different parts: **hypothesis** and **conclusion**.

Example



Rules

A rule contains two different parts: **hypothesis** and **conclusion**.

Example



"If x and y are co-workers, and y and z are co-workers, then x and z are also co-workers"

$$\forall x, y, z \text{ co-worker}(x, y) \wedge \text{co-worker}(y, z) \rightarrow \text{co-worker}(x, z)$$

Example

Example

Facts:

$works\text{-}for(Mark, LIRMM) \wedge$
 $works\text{-}for(Travis, LIRMM) \wedge$
 $works\text{-}for(Tom, LIRMM) \wedge$
 $plays\text{-}for(Mark, Team\ A) \wedge$
 $plays\text{-}for(Travis, Team\ B) \wedge$
 $plays\text{-}for(Tom, Team\ C) \wedge$
 $is\text{-}a(Team\ A, SquashClub) \wedge$
 $is\text{-}a(Team\ B, RugbyClub) \wedge$
 $is\text{-}a(Team\ C, SquashClub) \wedge$

Rules:

$\forall x, y, z\ works\text{-}for(x, z) \wedge works\text{-}for(y, z) \rightarrow co\text{-}worker(x, y)$
 $\forall x, y\ plays\text{-}for(x, y) \wedge is\text{-}a(y, SquashClub) \rightarrow plays(x, Squash)$
 $\forall x, y, z\ plays(x, z) \wedge plays(y, z) \rightarrow same\text{-}sport(x, y)$

Q1: $\exists x\ plays\text{-}for(x, Team\ B)$

Q2: $\exists x, y\ co\text{-}worker(x, y) \wedge same\text{-}sport(x, y)$

Example

Example

Facts:

$works\text{-}for(Mark, LIRMM) \wedge$
 $works\text{-}for(Travis, LIRMM) \wedge$
 $works\text{-}for(Tom, LIRMM) \wedge$
 $plays\text{-}for(Mark, Team\ A) \wedge$
 $plays\text{-}for(Travis, Team\ B) \wedge$
 $plays\text{-}for(Tom, Team\ C) \wedge$
 $is\text{-}a(Team\ A, SquashClub) \wedge$
 $is\text{-}a(Team\ B, RugbyClub) \wedge$
 $is\text{-}a(Team\ C, SquashClub) \wedge$

Rules:

$\forall x, y, z\ works\text{-}for(x, z) \wedge works\text{-}for(y, z) \rightarrow co\text{-}worker(x, y)$
 $\forall x, y\ plays\text{-}for(x, y) \wedge is\text{-}a(y, SquashClub) \rightarrow plays(x, Squash)$
 $\forall x, y, z\ plays(x, z) \wedge plays(y, z) \rightarrow same\text{-}sport(x, y)$

Q1: $\exists x\ plays\text{-}for(x, Team\ B)$

Answers: $\{(x, Travis)\}$

Q2: $\exists x, y\ co\text{-}worker(x, y) \wedge same\text{-}sport(x, y)$

Queries and rule application

Q2: $\exists x, y \text{ co-worker}(x, y) \wedge \text{same-sport}(x, y)$

R1: $\forall x, y, z \text{ works-for}(x, z) \wedge \text{works-for}(y, z) \rightarrow \text{co-worker}(x, y)$

R2: $\forall x, y \text{ plays-for}(x, y) \wedge \text{is-a}(y, \text{SquashClub}) \rightarrow \text{plays}(x, \text{Squash})$

R3: $\forall x, y, z \text{ plays}(x, z) \wedge \text{plays}(y, z) \rightarrow \text{same-sport}(x, y)$

Fact

works-for(Mark, LIRMM)

works-for(Travis, LIRMM)

works-for(Tom, LIRMM)

plays-for(Mark, Team A)

plays-for(Travis, Team B)

plays-for(Tom, Team C)

is-a(Team A, SquashClub)

is-a(Team B, RugbyClub)

is-a(Team C, SquashClub)

Queries and rule application

Q2: $\exists x, y \text{ co-worker}(x, y) \wedge \text{same-sport}(x, y)$

R1: $\forall x, y, z \text{ works-for}(x, z) \wedge \text{works-for}(y, z) \rightarrow \text{co-worker}(x, y)$

R2: $\forall x, y \text{ plays-for}(x, y) \wedge \text{is-a}(y, \text{SquashClub}) \rightarrow \text{plays}(x, \text{Squash})$

R3: $\forall x, y, z \text{ plays}(x, z) \wedge \text{plays}(y, z) \rightarrow \text{same-sport}(x, y)$

Fact

works-for(Mark, LIRMM)

works-for(Travis, LIRMM)

works-for(Tom, LIRMM)

plays-for(Mark, Team A)

plays-for(Travis, Team B)

plays-for(Tom, Team C)

is-a(Team A, SquashClub)

is-a(Team B, RugbyClub)

is-a(Team C, SquashClub)

co-worker(Mark, Travis)

co-worker(Mark, Tom)

co-worker(Travis, Mark)

co-worker(Travis, Tom)

co-worker(Tom, Mark)

co-worker(Tom, Travis)

Queries and rule application

Q2: $\exists x, y \text{ co-worker}(x, y) \wedge \text{same-sport}(x, y)$

R1: $\forall x, y, z \text{ works-for}(x, z) \wedge \text{works-for}(y, z) \rightarrow \text{co-worker}(x, y)$

R2: $\forall x, y \text{ plays-for}(x, y) \wedge \text{is-a}(y, \text{SquashClub}) \rightarrow \text{plays}(x, \text{Squash})$

R3: $\forall x, y, z \text{ plays}(x, z) \wedge \text{plays}(y, z) \rightarrow \text{same-sport}(x, y)$

Fact

works-for(Mark, LIRMM)

works-for(Travis, LIRMM)

works-for(Tom, LIRMM)

plays-for(Mark, Team A)

plays-for(Travis, Team B)

plays-for(Tom, Team C)

is-a(Team A, SquashClub)

is-a(Team B, RugbyClub)

is-a(Team C, SquashClub)

co-worker(Mark, Travis)

co-worker(Mark, Tom)

co-worker(Travis, Mark)

co-worker(Travis, Tom)

co-worker(Tom, Mark)

co-worker(Tom, Travis)

plays(Mark, Squash)

plays(Tom, Squash)

Queries and rule application

Q2: $\exists x, y \text{ co-worker}(x, y) \wedge \text{same-sport}(x, y)$

R1: $\forall x, y, z \text{ works-for}(x, z) \wedge \text{works-for}(y, z) \rightarrow \text{co-worker}(x, y)$

R2: $\forall x, y \text{ plays-for}(x, y) \wedge \text{is-a}(y, \text{SquashClub}) \rightarrow \text{plays}(x, \text{Squash})$

R3: $\forall x, y, z \text{ plays}(x, z) \wedge \text{plays}(y, z) \rightarrow \text{same-sport}(x, y)$

Fact

works-for(Mark, LIRMM)

works-for(Travis, LIRMM)

works-for(Tom, LIRMM)

plays-for(Mark, Team A)

plays-for(Travis, Team B)

plays-for(Tom, Team C)

is-a(Team A, SquashClub)

is-a(Team B, RugbyClub)

is-a(Team C, SquashClub)

co-worker(Mark, Travis)

co-worker(Mark, Tom)

co-worker(Travis, Mark)

co-worker(Travis, Tom)

co-worker(Tom, Mark)

co-worker(Tom, Travis)

plays(Mark, Squash)

plays(Tom, Squash)

same-sport(Mark, Tom)

same-sport(Tom, Mark)

Queries and rule application

Q2: $\exists x, y \text{ co-worker}(x, y) \wedge \text{same-sport}(x, y)$

R1: $\forall x, y, z \text{ works-for}(x, z) \wedge \text{works-for}(y, z) \rightarrow \text{co-worker}(x, y)$

R2: $\forall x, y \text{ plays-for}(x, y) \wedge \text{is-a}(y, \text{SquashClub}) \rightarrow \text{plays}(x, \text{Squash})$

R3: $\forall x, y, z \text{ plays}(x, z) \wedge \text{plays}(y, z) \rightarrow \text{same-sport}(x, y)$

Fact

works-for(Mark, LIRMM)

works-for(Travis, LIRMM)

works-for(Tom, LIRMM)

plays-for(Mark, Team A)

plays-for(Travis, Team B)

plays-for(Tom, Team C)

is-a(Team A, SquashClub)

is-a(Team B, RugbyClub)

is-a(Team C, SquashClub)

co-worker(Mark, Travis)

co-worker(Mark, Tom)

co-worker(Travis, Mark)

co-worker(Travis, Tom)

co-worker(Tom, Mark)

co-worker(Tom, Travis)

plays(Mark, Squash)

plays(Tom, Squash)

same-sport(Mark, Tom)

same-sport(Tom, Mark)

Answers: $\{(x, \text{Mark}), (y, \text{Tom})\}$ & $\{(x, \text{Tom}), (y, \text{Mark})\}$

Goals & Challenges

This problem has been widely studied before. However, new factors have appeared to change its nature.

Goals & Challenges

This problem has been widely studied before. However, new factors have appeared to change its nature.

New challenge: Large knowledge bases

- F can now be very **large** (see the Semantic Web)
- Large \rightarrow Does not fit in main memory.

Goals & Challenges

This problem has been widely studied before. However, new factors have appeared to change its nature.

New challenge: Large knowledge bases

- F can now be very **large** (see the Semantic Web)
- Large \rightarrow Does not fit in main memory.

New question: **“How can we have efficiently an answer to Q , when \mathcal{F} is very large?”**

Thesis objectives

Three different approaches to this problem exist in the litterature:

- 1 Approximative and probabilistic algorithms.
- 2 Algorithms optimization.
- 3 Analysis of storage methods.

Thesis objectives

Three different approaches to this problem exist in the literature:

- 1 Approximative and probabilistic algorithms.
- 2 Algorithms optimization.
- 3 Analysis of storage methods.

We suspect (and hope to prove) items 2 and 3 are tightly correlated. Our objectives for the first year of thesis were:

- Investigating different storage models (RDBs, GDBs & Triple Stores) and their storage and querying efficiency.
- Programming a backtrack algorithm in order to compute homomorphisms.
- Using this algorithm to perform Ontological Conjunctive Query Answering over those storage systems.

Table of Contents

- 1 Research problem
- 2 Encodings & Translations**
- 3 Preliminary results
- 4 Future work
- 5 Questions

Encoding: Fact \rightarrow Set

Encoding the fact from our example:

Encoding: Fact \rightarrow Set

Encoding the fact from our example:

$\{$ *works-for*(Mark, LIRMM), *works-for*(Travis, LIRMM),
works-for(Tom, LIRMM), *plays-for*(Mark, Team A), *plays-for*(Travis, Team B),
plays-for(Tom, Team C), *is-a*(Team A, SquashClub),
is-a(Team B, RugbyClub), *is-a*(Team C, SquashClub) $\}$

- Encoded yes, however totally unstructured.
- The complexity of every atomic operation depend on the size of the knowledge base in atoms.

Encoding: Fact \rightarrow Tables

Structuring our fact by the atoms predicates, we obtain **tables**:

Encoding: Fact \rightarrow Tables

Structuring our fact by the atoms predicates, we obtain **tables**:

works-for	
1	2
<i>Mark</i>	<i>LIRMM</i>
<i>Travis</i>	<i>LIRMM</i>
<i>Tom</i>	<i>LIRMM</i>

plays-for	
1	2
<i>Mark</i>	<i>Team A</i>
<i>Travis</i>	<i>Team B</i>
<i>Tom</i>	<i>Team C</i>

is-a	
1	2
<i>Team A</i>	<i>SquashClub</i>
<i>Team B</i>	<i>RugbyClub</i>
<i>Team C</i>	<i>SquashClub</i>

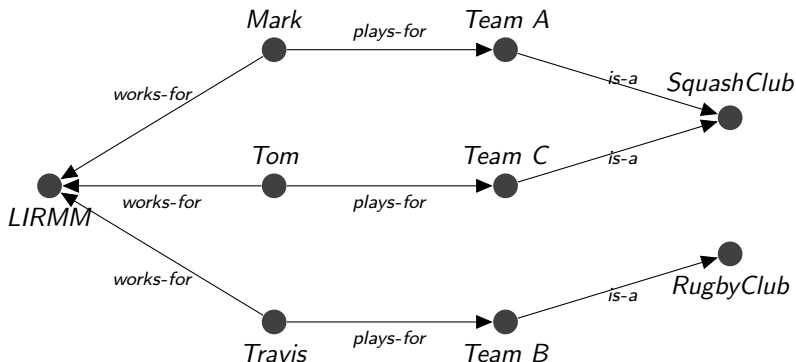
- This encoding can be directly stored in a Relational Database.
- Querying is then available either with BackTrack, either with a SQL interface.

Encoding: Fact \rightarrow Graph

Structuring the fact, this time by its terms, we obtain a **graph**:

Encoding: Fact \rightarrow Graph

Structuring the fact, this time by its terms, we obtain a **graph**:



Analysis

- Encoding a fact without a structure is totally inappropriate for our problem.

Analysis

- Encoding a fact without a structure is totally inappropriate for our problem.
- Relational Databases handle very well knowledge located in secondary memory, however:
 - Atomic operations of the BackTrack use SQL operations which complexity also depend on the size of the tables.
 - Using SQL instead may also not be the best solution: Joins become very costly as the number of predicates increases.

Analysis

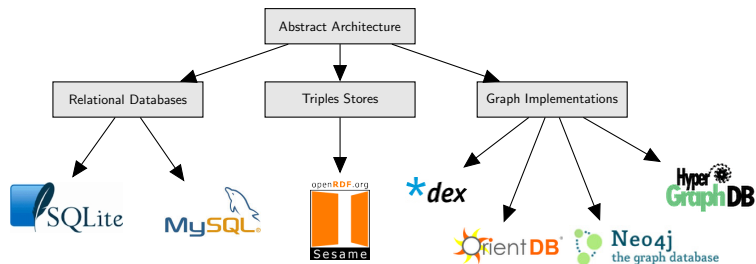
- Encoding a fact without a structure is totally inappropriate for our problem.
- Relational Databases handle very well knowledge located in secondary memory, however:
 - Atomic operations of the BackTrack use SQL operations which complexity also depend on the size of the tables.
 - Using SQL instead may also not be the best solution: Joins become very costly as the number of predicates increases.
- Running the BackTrack algorithm with a graph works very well when the graph is stored in main memory. Unfortunately, we do not know how to do it when it is not the case.

Table of Contents

- 1 Research problem
- 2 Encodings & Translations
- 3 Preliminary results**
- 4 Future work
- 5 Questions

Chosen tools

We have selected a set of storage tools to proceed with our tests:



Next step: How to use all those different tools for the same purpose?

Alaska project

Alaska Project:

Abstract **L**ogic-based **A**rchitecture for **S**torage systems & **K**nowledge bases **A**nalysis

- Implementation of classes and interfaces that ensure that all the storage systems plugged in will answer to the same methods using a common type of data.
- Written in JAVA: Very easy to plug several pieces of code in, however, with a significant loss in speed and efficiency.

Alaska: Architecture

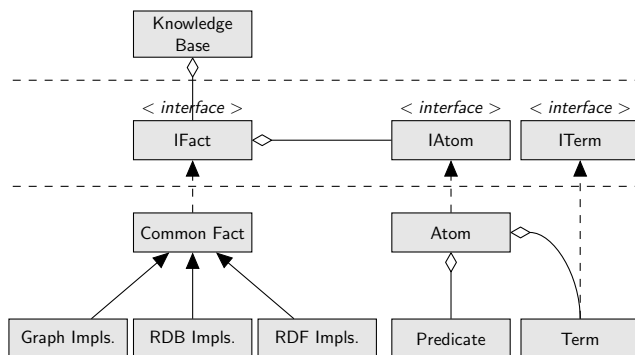


Figure: Class diagram for the architecture.

IFact methods

Below the list of the elementary methods that a class implementing the IFact interface must have defined:

- To add a new term.
- To add a new atom.
- To list all terms.
- To list all atoms.
- To retrieve a term by its label.
- To indicate whether there is an atom containing two given terms.

Storage tests

Using our platform, we have evaluated the insertion efficiency of different storage systems:

$$\mathcal{F} \models Q$$

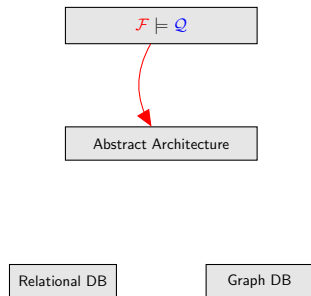
Abstract Architecture

Relational DB

Graph DB

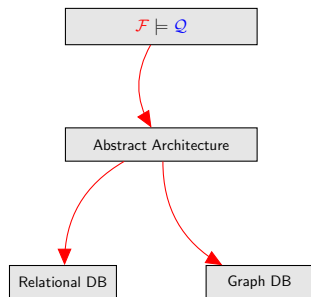
Storage tests

Using our platform, we have evaluated the insertion efficiency of different storage systems:



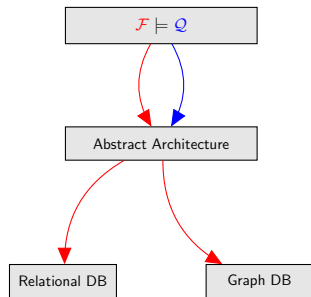
Storage tests

Using our platform, we have evaluated the insertion efficiency of different storage systems:



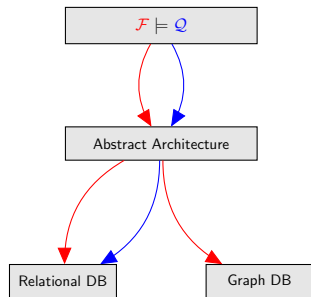
Storage tests

Using our platform, we have evaluated the insertion efficiency of different storage systems:



Storage tests

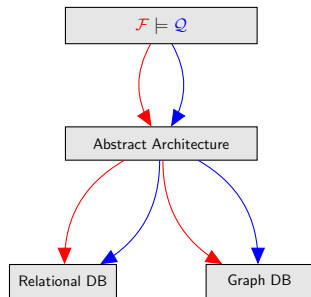
Using our platform, we have evaluated the insertion efficiency of different storage systems:



Comparison of the insertion test results				
–	DEX	HyperGraphDB	Sqlite	MySQL
1k	1 s	3 s	5 s	4 s
5k	1 s	6 s	7 s	13 s
15k	2 s	10 s	57 s	54 s
25k	2 s	17 s	185 s	130 s
40k	4 s	48 s	505 s	316 s
60k	6 s	177 s	1131 s	674 s
75k	7 s	322 s	1779 s	1038 s
100k	10 s	601 s	3226 s	1790 s

Storage tests

Using our platform, we have evaluated the insertion efficiency of different storage systems:



–	DEX	HyperGraphDB	Sqlite	MySQL
1k	1 s	3 s	5 s	4 s
5k	1 s	6 s	7 s	13 s
15k	2 s	10 s	57 s	54 s
25k	2 s	17 s	185 s	130 s
40k	4 s	48 s	505 s	316 s
60k	6 s	177 s	1131 s	674 s
75k	7 s	322 s	1779 s	1038 s
100k	10 s	601 s	3226 s	1790 s

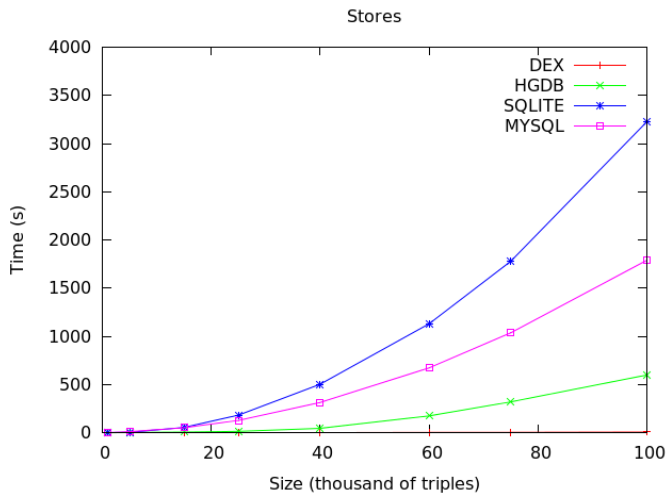


Table of Contents

- 1 Research problem
- 2 Encodings & Translations
- 3 Preliminary results
- 4 Future work**
- 5 Questions

Completing insertion tests

We still have to perform the insertion tests with Sesame triple store.

Completing this part of the work will enable us to start running querying performances tests.

- 1 month: Finding a solution for the insertion of logical atoms in a Triples Store.
- 1 month: Running the tests and comparing the efficiency of the Store against other storage systems.

Total length: 2 months

Evaluation of querying engines

Performing querying tests will be very important in order to define precisely the questions this thesis will address in details.

- 1 month: Performing all the queries.
- 1 month: Analysis of the results and evaluation of the necessity to change our set of queries.
- 1 month: Definition a final set of queries for the following steps.
- 1 month: Comparisons between the built-in querying languages and SQL via our DatalogToSQL algorithm.
- 1 month: Analysis and dissemination of the results obtained, publication of the results.

Total length: 5 months

Querying test

With our platform + our DatalogToSQL algorithm, we aim evaluating querying performances of the selected storage systems:

$$\mathcal{F} \models Q$$

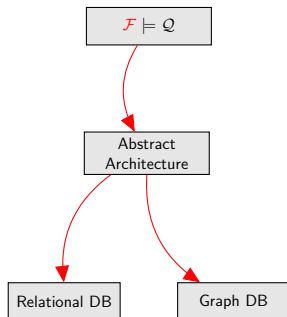
Abstract
Architecture

Relational DB

Graph DB

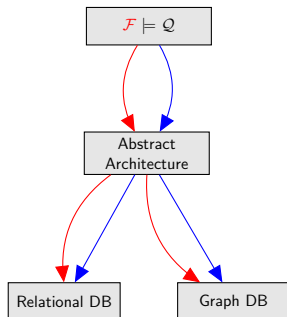
Querying test

With our platform + our DatalogToSQL algorithm, we aim evaluating querying performances of the selected storage systems:



Querying test

With our platform + our DatalogToSQL algorithm, we aim evaluating querying performances of the selected storage systems:

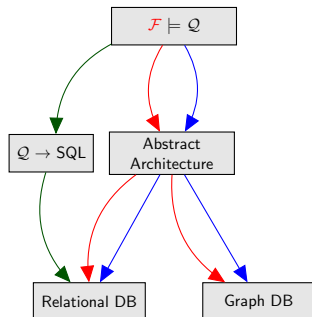


Test results		
–	Query size	Querying time
<i>BT</i>	<i>... terms</i>	<i>... ms</i>

Test results		
–	Query size	Querying time
<i>BT</i>	<i>... terms</i>	<i>... ms</i>

Querying test

With our platform + our DatalogToSQL algorithm, we aim evaluating querying performances of the selected storage systems:

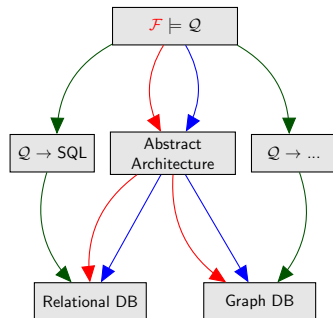


Test results		
–	Query size	Querying time
<i>BT</i>	... terms	... ms
<i>SQL</i>	... terms	... ms

Test results		
–	Query size	Querying time
<i>BT</i>	... terms	... ms

Querying test

With our platform + our DatalogToSQL algorithm, we aim evaluating querying performances of the selected storage systems:



Test results		
–	Query size	Querying time
<i>BT</i>	<i>... terms</i>	<i>... ms</i>
<i>SQL</i>	<i>... terms</i>	<i>... ms</i>

Test results		
–	Query size	Querying time
<i>BT</i>	<i>... terms</i>	<i>... ms</i>
<i>Graph</i>	<i>... terms</i>	<i>... ms</i>

Backtrack optimization

Optimizing our Backtrack algorithm is planned in the second part of the upcoming year:

- **1 month:** Reading and searching if other techniques exist beside the ones we know.
- **1 month:** Understanding those techniques and implementing them within our framework.
- **1 month:** Checking their correctness and debugging if necessary.
- **1 month:** Evaluating the performance of those techniques against the results we already have.
- **1 month:** Analysis and dissemination of the results obtained, publication of the results.

Total length: 5 months

Large knowledge bases

In parallel, we aim extending our work to large KBs:

- **3 months:** Identifying large knowledge sets in use in research and picking those that seem more relevant to our study.
- **1 month:** Comparing of the previous test results previously obtained against tests with real knowledge bases.
- **2 months:** Understanding the results obtained and identifying possible causes of poor efficiency with those knowledge bases.
- **3 months:** Developing the necessary changes in order to consolidate our platform allowing it to work exclusively with those bases.
- **1 month:** Analysis and dissemination of the results obtained, publication of the results.

Total length: 10 months

Table of Contents

- 1 Research problem
- 2 Encodings & Translations
- 3 Preliminary results
- 4 Future work
- 5 Questions**

Questions

Thank you!

Questions & comments...