

Ontological Conjunctive Query Answering over Large Knowledge Bases

Bruno Paiva Lima da Silva, Jean-François Baget, Madalina Croitoru
{bplsilva,baget,croitoru}@lirmm.fr

Université Montpellier 2

April 16, 2011

- 1 Research problem
- 2 Encodings & Translations
- 3 Current work
- 4 Conclusion
- 5 Questions

Table of Contents

- 1 Research problem
- 2 Encodings & Translations
- 3 Current work
- 4 Conclusion
- 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 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 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**)

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"

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

Rules semantics are that anytime the hypothesis of a rule is found in the facts, its conclusion is then added to the KB as new information.

Finding substitutions

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

Finding substitutions

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

We focus our work on finding substitutions between terms from a given query (constants or variables) and the terms from our facts.

In order to do it, we use a **BackTrack** algorithm.

Different methods for KR and manipulation by dedicated reasoning systems have been successfully studied in the past.

Goals & Challenges

We focus our work on finding substitutions between terms from a given query (constants or variables) and the terms from our facts.

In order to do it, we use a **BackTrack** algorithm.

Different methods for KR and manipulation by dedicated reasoning systems have been successfully studied in the past.

Large knowledge bases: New challenge

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

Goals & Challenges

We focus our work on finding substitutions between terms from a given query (constants or variables) and the terms from our facts.

In order to do it, we use a **BackTrack** algorithm.

Different methods for KR and manipulation by dedicated reasoning systems have been successfully studied in the past.

Large knowledge bases: New challenge

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

“Can we have efficiently an answer to Q , when \mathcal{F} is very large?”

Table of Contents

- 1 Research problem
- 2 Encodings & Translations**
- 3 Current work
- 4 Conclusion
- 5 Questions

Encoding: Fact \rightarrow Set

Encoding the fact from our example:

Encoding: Fact \rightarrow Set

Encoding the fact from our example:

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

- 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		plays-for		is-a	
1	2	1	2	1	2
<i>Mark</i>	<i>LIRMM</i>	<i>Mark</i>	<i>Team A</i>	<i>Team A</i>	<i>SquashClub</i>
<i>Travis</i>	<i>LIRMM</i>	<i>Travis</i>	<i>Team B</i>	<i>Team B</i>	<i>RugbyClub</i>
<i>Tom</i>	<i>LIRMM</i>	<i>Tom</i>	<i>Team C</i>	<i>Team C</i>	<i>SquashClub</i>

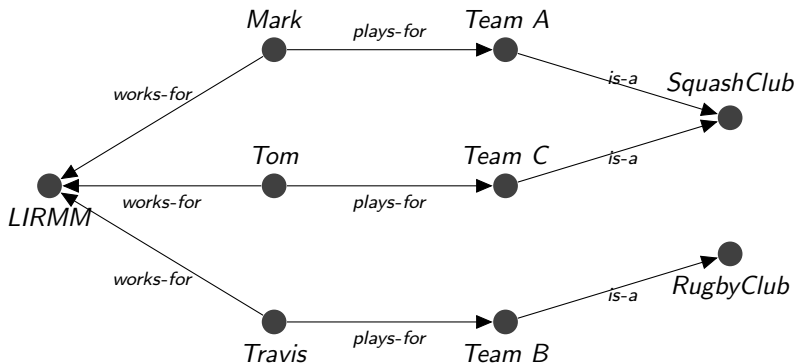
- This encoding can be directly stored in a Relational Database.
- Querying is then available either with BackTrack, either with a SQL interface.

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, it does not scale very well.

Table of Contents

- 1 Research problem
- 2 Encodings & Translations
- 3 Current work**
- 4 Conclusion
- 5 Questions

Current challenges

In order to be able to perform reasoning over very large knowledge bases, we started searching for storage systems:

Current challenges

In order to be able to perform reasoning over very large knowledge bases, we started searching for storage systems:

- that have the ability to support very large knowledge bases stored in secondary memory.

Current challenges

In order to be able to perform reasoning over very large knowledge bases, we started searching for storage systems:

- that have the ability to support very large knowledge bases stored in secondary memory.
- efficient on homomorphism elementar operations, such as:
 - computing & retrieving the neighbourhood of a term and to be able to iterate over this structure.
 - checking whether there is a given relation between two given nodes or not.

Current challenges

In order to be able to perform reasoning over very large knowledge bases, we started searching for storage systems:

- that have the ability to support very large knowledge bases stored in secondary memory.
- efficient on homomorphism elementar operations, such as:
 - computing & retrieving the neighbourhood of a term and to be able to iterate over this structure.
 - checking whether there is a given relation between two given nodes or not.
- in which the complexity (time) of the insertion of a new atom does not depend on the size of the KB.

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

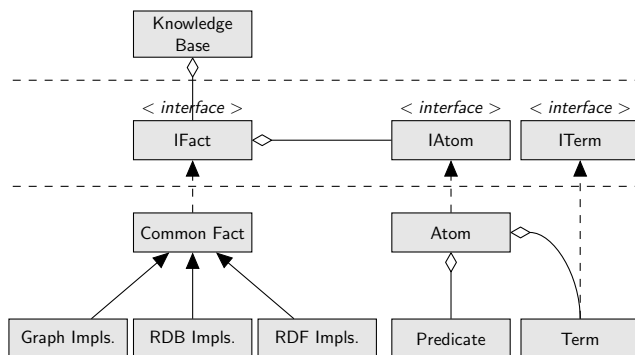


Figure: Class diagram for the architecture.

Application #1

Comparing storage systems between themselves:

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

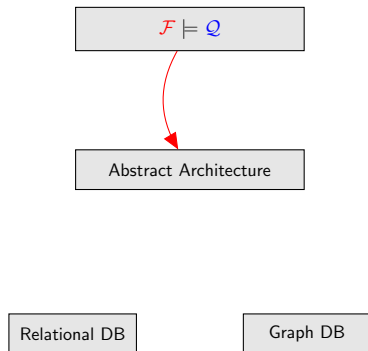
Abstract Architecture

Relational DB

Graph DB

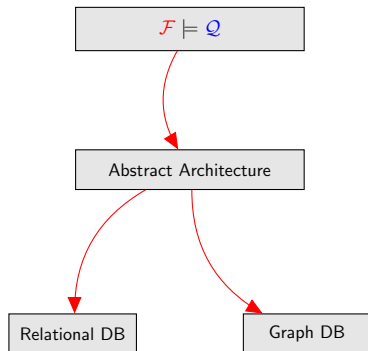
Application #1

Comparing storage systems between themselves:



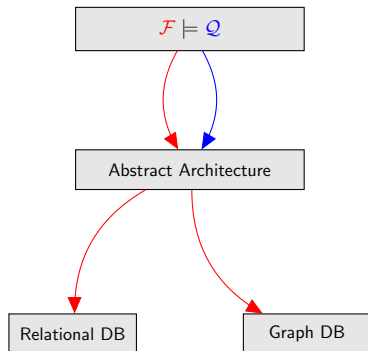
Application #1

Comparing storage systems between themselves:



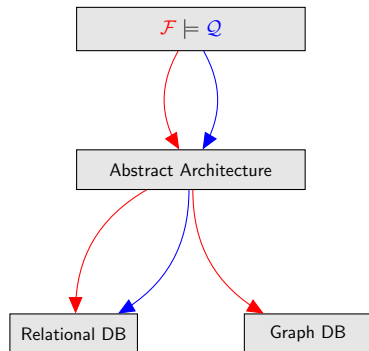
Application #1

Comparing storage systems between themselves:



Application #1

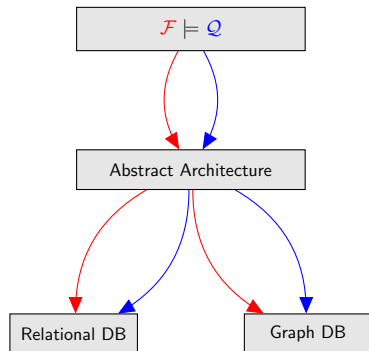
Comparing storage systems between themselves:



Test results		
<i>Name</i>	<i>KB size</i>	<i>Querying time</i>
<i>RDB</i>	<i>... Mb</i>	<i>... ms</i>

Application #1

Comparing storage systems between themselves:



Test results		
<i>Name</i>	<i>KB size</i>	<i>Querying time</i>
<i>RDB</i>	<i>... Mb</i>	<i>... ms</i>
<i>GDB</i>	<i>... Mb</i>	<i>... ms</i>

Application #2

Comparing different querying interfaces for a same storage system:

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

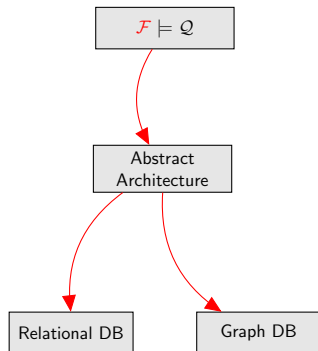
Abstract
Architecture

Relational DB

Graph DB

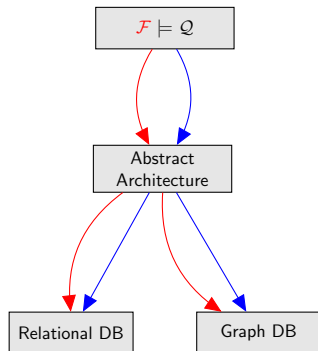
Application #2

Comparing different querying interfaces for a same storage system:



Application #2

Comparing different querying interfaces for a same storage system:

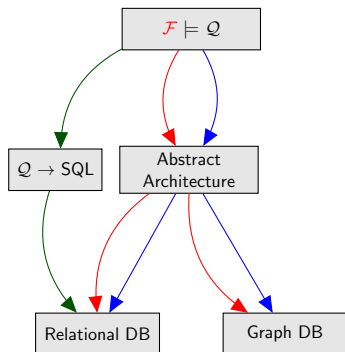


Test results		
–	Query size	Querying time
<i>BT</i>	<i>... terms</i>	<i>... ms</i>

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

Application #2

Comparing different querying interfaces for a same storage system:

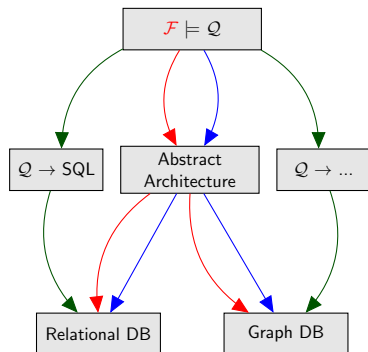


Test results		
–	Query size	Querying time
<i>BT</i>	... terms	... ms
<i>SQL</i>	... terms	... ms

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

Application #2

Comparing different querying interfaces for a same storage system:

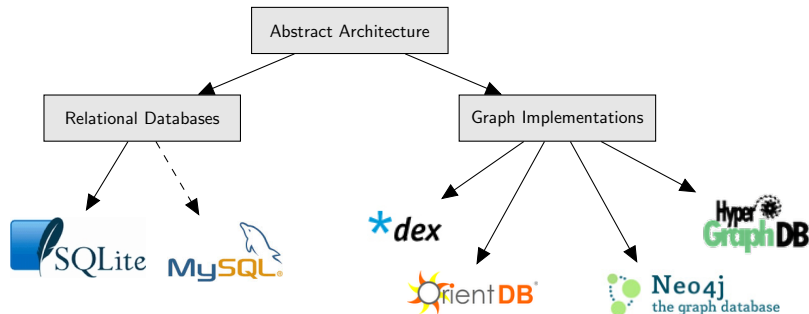


Test results		
–	Query size	Querying time
<i>BT</i>	... terms	... ms
<i>SQL</i>	... terms	... ms

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

Implementations

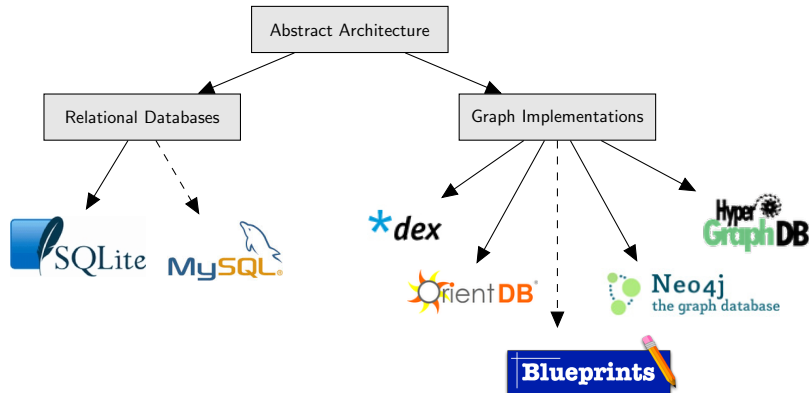
Implementations currently supported by the Alaska project.



Next step: Which kind of data to use?

Implementations

Implementations currently supported by the Alaska project.



Next step: Which kind of data to use?

Table of Contents

- 1 Research problem
- 2 Encodings & Translations
- 3 Current work
- 4 Conclusion**
- 5 Questions

Future work

As the execution performance also came into play in our research problem, our future work will consist in:

- finding and plugging more pertinent storage systems into our system.
- identifying any other problems that might have an influence when querying over large knowledge bases.
- running tests against several large knowledge bases available throughout the web.
- identifying the storage methods that answer best our problem, and where improvements can be made.

Then after...

We will also consider working on:

- implementing some kind of knowledge generator that would generate unbiased facts, which we could test against real data.
- optimizing our BackTrack algorithm in order to enhance the performance of our system.
- perhaps implementing a rule application system in order to tackle the RULE-ENTAILMENT problem.

Table of Contents

- 1 Research problem
- 2 Encodings & Translations
- 3 Current work
- 4 Conclusion
- 5 Questions**

Questions

Thank you!

Questions & comments...