Challenges & Issues Of an RSP-QL Implementation

Riccardo Tommasini and Emanuele Della Valle
Politecnico di Milano, DEIB, Milan, Italy
Who I AM

• Ph.D. Student at Politecnico di Milano

• Advisor: Emanuele Della Valle

• Work on:
  
  • RSP Benchmarking

  • Expressive Stream Reasoning
The Story

Why Implementing RSP-QL

RSP-QL is a reference model to explain the semantics of exiting RDF Stream Processing Engines.

Prototypes have a central role to support and foster Semantic Web and Stream Reasoning research.

The issues discovered realising system* enable more foundational science.
The Story

Looking behind the implementation curtains
The Theory

- RSP-QL is a **reference model** to explain the semantics of exiting RDF Stream Processing **Engines**.
- It extends SPARQL 1.1 to include **RDF Streams** and the necessary **operators** to process them.
- It extends SPARQL 1.1 **evaluation** to include time.
The Theory

RSP-QL in a Nutshell

(a) RDFStream

(b) window \((o_i, c_i]\)

(c) Time-varying graph

(d) Instantaneous graph
The Code

Yasper 1.0

Stream
Stream Item

Windowing
Window Operators

SDS
Instantaneous Graph

R2S
Continuous Execution

Quering
Query

Response
Formatter

Reasoning

Time-Varying Graph
The Code
Yasper 1.0

• Generic Data Model, e.g., Triples and Named Graphs
• Configurable Time Model:
  • Externally Controlled Time
  • Ingestion Time vs Event Time
• Multi Stream Queries and SDS Consolidation & Maintenance
The Challenges

• (Q1) What is the best way to model the stream content?
• (Q2) How does the time model impact the processing?
• (Q3) What does define the current SDS at time t?
• (Q4) Is there an efficient way to maintain the SDS?
Generic Data Model

- Enables optimisation related to the query language
- Requires query rewriting to reduce overhead

RDF Triples

- Adopted my the majority of the existing RSP Engines
- Lower latency on simple queries

Named Graphs

- RSP-QL model of choice
- Better throughput on complex queries
Stream Ordering

• In a **web environment**, which is distributed and federated, **out-of-order arrivals matters**

• **RSP-QL partially relaxed** the assumption of totally ordered RDF stream

• **Event-time vs Ingestion-time** might influence the correctness of the RSP Engine
SDS Consolidation & Maintenance

• **Assumption**: Instantaneous Graph can be generated by a Time-Varying Graph for any instant $t$ for which a window operator is defined.

• **Consequence 1**: the SDS content is always up-to-date

• **Consequence 2**: the RSP engine to maintain sub-windows accessible.
SDS Consolidation & Maintenance

(a) WINDOW [ RANGE 5s STEP 5s]
(b) WINDOW [ RANGE 5s STEP 2s]

<table>
<thead>
<tr>
<th>SDS(30)</th>
<th>W1</th>
<th>W2</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td></td>
<td>y1,y2</td>
</tr>
<tr>
<td>SDS(35)</td>
<td>x2,x3</td>
<td>y3,y4</td>
</tr>
<tr>
<td>SDS(36)</td>
<td>x4</td>
<td>y4</td>
</tr>
</tbody>
</table>
Closed Active SDS

It considers only the instantaneous RDF graphs that were derived from closed active windows.
Cached Active SDS

It **considers** the instantaneous RDF graphs from the **latest** closed active window, which was **cached**.

(a) WINDOW [ RANGE 5s STEP 5s]
(b) WINDOW [ RANGE 5s STEP 2s]

<table>
<thead>
<tr>
<th></th>
<th>W1</th>
<th>W2</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDS(30)</td>
<td>x1</td>
<td></td>
</tr>
<tr>
<td>SDS(35)</td>
<td>x2, x3</td>
<td>y3, y4</td>
</tr>
<tr>
<td>SDS(36)</td>
<td>x2, x3</td>
<td>y3, y4</td>
</tr>
</tbody>
</table>
The SDS Maintenance

• Deciding the semantic of the active SDS is not the only problem.

• How do we maintain the SDS consistent with frequent updates might influence the performance.

• Can I do it efficiently (incrementally)?
Opportunity: Descriptive Study

• Use Architectural Variants to define features for a descriptive study

• Generate a set of statistical hypothesis out of the scientific Hypotheses we formulated for each challenge (In the paper)
Opportunity: Syntax

- RSP-QL is very expressive
- Maybe too expressive to be captured by a single declarative QL.
- **Proposal**: a syntax which is more usable yet less expressive.
• Can you fill this slide for me?