Naive reasoning on RDF streams

Emanuele Della Valle
emanuele.dellavalle@polimi.it
http://emanueledellavalle.org
Share, Remix, Reuse — Legally

- This work is licensed under the Creative Commons Attribution 3.0 Unported License.

- **Your are free:**
  - **to Share** — to copy, distribute and transmit the work
  - **to Remix** — to adapt the work

- **Under the following conditions**
  - **Attribution** — You must attribute the work by inserting
    - “[source http://streamreasoning.org/sr4ld2013]” at the end of each reused slide
    - a credits slide stating
      - These slides are partially based on “Streaming Reasoning for Linked Data 2013” by M. Balduini, J-P Calbimonte, O. Corcho, D. Dell'Aglio, E. Della Valle, and J.Z. Pan http://streamreasoning.org/sr4ld2013

- To view a copy of this license, visit [http://creativecommons.org/licenses/by/3.0/](http://creativecommons.org/licenses/by/3.0/)
Agenda

- Forms of reasoning for Q/A
- Naïve forms of stream reasoning for Q/A
- A not so naïve form of stream reasoning for Q/A
- Wrap up
MEMO: forms of reasoning for Q/A

- Data-driven (a.k.a. forward reasoning)
  - RDF data \(\rightarrow\) Reasoner \(\rightarrow\) Inferred data \(\rightarrow\) SPARQL
  - Ontology

- Query-driven – backward reasoning
  - RDF data \(\rightarrow\) Reasoner \(\rightarrow\) SPARQL
  - Ontology

- Query-driven – query rewriting (a.k.a. ontology based data access)
  - Data \(\leftrightarrow\) Rewritten query \(\leftrightarrow\) Reasoner \(\leftrightarrow\) SPARQL
  - Ontology
Naïve Stream Reasoning

- Data-driven (a.k.a. forward reasoning)
  - S2R → RDF data → Reasoner → Inferred data → SPARQL
  - RDF data
  - Reasoner
  - Inferred data
  - SPARQL
  - ontology

- Query-driven – backward reasoning
  - S2R → RDF data → Reasoner → SPARQL
  - RDF data
  - Reasoner
  - SPARQL
  - ontology

- Query-driven – query rewriting (a.k.a. ontology based data access)
  - S2R → data → Rewritten query → Reasoner → SPARQL
  - data
  - Rewritten query
  - Reasoner
  - SPARQL
  - ontology
Example of Stream Reasoning 1/2

- Query: measure the impact of Alice's microposts
MEMO: our running example data model

- For example

Alice posts \( p_1 \).

Bob posts \( p_2 \).

\( p_1 \) discusses \( p_2 \).

\( p_2 \) discusses \( p_4 \) and \( p_7 \).

\( p_4 \) discusses \( p_7 \).

\( p_6 \) discusses \( p_8 \).

The timeline also shows the timestamps for each post:

- 50 min ago: \( p_1 \)
- 40 min ago: \( p_2 \)
- 30 min ago: \( p_3 \)
- 20 min ago: \( p_4 \)
- 10 min ago: \( p_5 \)
- Now: \( p_6 \), \( p_7 \), and \( p_8 \)
What impact has been my micropost $p_1$ creating in the last hour? Let’s count the number of microposts that discuss it …

REGISTER STREAM ImpactMeter AS
SELECT (count(?p) AS ?impact)
FROM STREAM <http://.../fb> [RANGE 60m STEP 10m]
WHERE {
  :Alice posts [ sr:discusses ?p ]
}

Transitive property

Alice posts $p_1$.
Naïve data-driven stream reasoning

**Memo**

- S2R → RDF data → Reasoner → Inferred data → SPARQL

- $p_1$

- now

http://streamreasoning.org/sr4ld2013

ISWC 2013
Sydney, Australia
Naïve data-driven stream reasoning

- **Memo**

  S2R → RDF data → Reasoner → Inferred data → SPARQL

  \[ p_1 \rightarrow \text{discusses} \rightarrow p_2 \]

  10 min ago → now
Naïve data-driven stream reasoning

- **Memo**

  - S2R → RDF data → Reasoner → Inferred data → SPARQL
  - ontology

  - \( p_1 \) discusses \( p_2 \)
  - \( p_2 \) discusses \( p_3 \)

  - Timeline:
    - 20 min ago
    - 10 min ago
    - now
Naïve data-driven stream reasoning

MEMO

S2R → RDF data → Reasoner → Inferred data → SPARQL

ontology

m1 discusses m2

m2 discusses m4

m4 discusses m1

m3 discusses m2

30 min ago → 20 min ago → 10 min ago → now

http://streamreasoning.org/sr4ld2013
The entire inference process is repeated each time the SR2 operator delivers new RDF data.
Naïve data-driven stream reasoning

- Memo

```
S2R -> RDF data -> Reasoner -> Inferred data

ontology

p1 \(\text{discusses}\) p2 \(\text{discusses}\) p4 \(\text{discusses}\) p7

p2 \(\text{discusses}\) p3 \(\text{discusses}\) p5 \(\text{discusses}\) p8

p3 \(\text{discusses}\) p6

p4 \(\text{discusses}\) p5

50 min ago 40 min ago 30 min ago 20 min ago 10 min ago now
```

http://streamreasoning.org/sr4ld2013
Naïve data-driven stream reasoning

- Memo

S2R → RDF data → Reasoner → Inferred data → SPARQL

- Memes

p1 discusses p2
p2 discusses p3
p3 discusses p4
p4 discusses p5
p5 discusses p6
p6 discusses p7
p7 discusses p8
p8 discusses p9

50 min ago 40 min ago 30 min ago 20 min ago 10 min ago now
Naïve data-driven stream reasoning

**Memo**

The reasoner infers data that is irrelevant to query answering.
Naive query-driven (backward) stream reasoning

- Memo

S2R → RDF data ← Reasoner ← SPARQL

- ontology

$p_1$

now
Naïve query-driven (backward) stream reasoning

Memo

- S2R → RDF data → Reasoner → Inferred data → SPARQL

- 10 min ago
- now

- \( p_1 \) discusses \( p_2 \)

- ontology
Naïve query-driven (backward) stream reasoning

- Memo

```
S2R → RDF data → Reasoner → Inferred data → SPARQL
```

- 2

Reasoner

Ontology

```
p1 discusses p2
p2 discusses p3
p3 discusses p1
```

Timeline:
- 20 min ago
- 10 min ago
- Now
Naive query-driven (backward) stream reasoning

- Memo

1. S2R
2. RDF data
3. Reasoner
4. Inferred data
5. SPARQL
6. ontology

- Diagram:
  - p_1 discusses p_2
  - p_2 discusses p_3
  - p_1 discusses p_3
  - p_2 discusses p_4

- Timeline:
  - 30 min ago
  - 20 min ago
  - 10 min ago
  - now

- Naïve query-driven (backward) stream reasoning

http://streamreasoning.org/sr4ld2013
Naive query-driven (backward) stream reasoning

**Memo**

The entire inference process is repeated each time the SR2 operator delivers new RDF data.
Naive query-driven (backward) stream reasoning

- Memo

S2R → RDF data → Reasoner → Inferred data → SPARQL

Ontology

p1 discusses p2
p2 discusses p3
p3 discusses p4
p4 discusses p5
p5 discusses p6

50 min ago 40 min ago 30 min ago 20 min ago 10 min ago now

http://streamreasoning.org/sr4ld2013
Naive query-driven (backward) stream reasoning

- **Memo**

  S2R → RDF data → Reasoner → Inferred data → SPARQL

  - 50 min ago
  - 40 min ago
  - 30 min ago
  - 20 min ago
  - 10 min ago
  - Now

  - p1
  - p2
  - p3
  - p4
  - p5
  - p6
  - p7
  - p8

  Discusses:
  - p1
  - p2
  - p3
  - p4
  - p5
  - p6
  - p7
  - p8

  Ontology:

  - Naïve query-driven (backward)
  - Stream reasoning

  - RDF data
  - Inferred data

  - S2R
  - SPARQL
Naive query-driven (backward) stream reasoning

- Memo

S2R → RDF data → Reasoner → Inferred data → SPARQL

ontology

- The backward reasoner would not even start!
MEMO

It is not that straightforward :-(

- Lack of a standard query language for DSMS and CEP
- Lack of a well-understood operational semantics for DSMS and CEP (cf. SECRET by I. Botan et al., PVLDB 3(1), 2010)
- Lack of expressiveness in OWL2QL
  - Temporal reasoning
  - Aggregates
  - Functions
Not so naïve stream reasoning

- Naïve data-driven approach

- From snapshots to changes
  - What has just been inserted?
  - What has just been deleted?

S2R → RDF data → Reasoner → Inferred data → SPARQL

Incremental!!!

S2R → insertions → Reasoner → Inferred data → SPARQL

S2R → deletions → Reasoner → Inferred data → SPARQL
Not so naïve stream reasoning

**MEMO**

The problem is that materialization (the result of data-driven processing) are very difficult to decrement efficiently.

- State-of-the-art: DRed algorithm
  - Over delete
  - Re-derive
  - Insert

- **Overestimation of deletion**: Overestimates deletions by computing all direct consequences of a deletion.

- **Rederivation**: Prunes those estimated deletions for which alternative derivations (via some other facts in the program) exist.

- **Insertion**: Adds the new derivations that are consequences of insertions to extensional predicates.
The Intuition of DRed Algorithm

- Let’s assume that we have the following materialized graph

- While inserts are not problematic, deletion are difficult to handle. If we delete \( p_2 \) discusses \( p_1 \) (\( p_2 \rightarrow p_1 \)), we have
  - **overestimate the impact of the deletion** and mark for deletion \( p_4 \rightarrow p_1 \) that can be derived by \( p_4 \rightarrow p_2 \) and \( p_2 \rightarrow p_1 \)
  - **look for alternative derivation** of \( p_4 \rightarrow p_1 \) and eventually find the chain \( p_4 \rightarrow p_3 \) and \( p_3 \rightarrow p_1 \)
Not so naïve data-driven stream reasoning

- Memo

S2R \[\text{insertions}\] Reasoner \[\text{Inferred data}\] SPARQL

\[\text{deletions}\]

Inferred data

ontology

Incremental !!!

\[p_1\]

now
Not so naïve data-driven stream reasoning

- Memo

S2R \[\text{insertions} \] Reasoner \[\text{Inferred data} \]

\[\text{deletions} \]

ontology

Incremental !!!

SPARQL

discusses $p_2$

$p_1$

10 min ago now

http://streamreasoning.org/sr4ld2013
Not so naïve data-driven stream reasoning

- Memo

S2R → Reasoner → Inferred data

insertions

deletions

ontology

Incremental!!!

SPARQL

Reasoner

S2R

p₁

discusses

p₂

discusses

p₃

20 min ago
10 min ago
now
Not so naïve data-driven stream reasoning

- Memo

S2R \[\text{insertions}\] Reasoner \[\text{Inferred data}\] SPARQL

\[\text{deletions}\]

Inferred data

ontology

Reasoner

S2R

\[\text{Incremental !!!}\]

\[\text{discusses}\]

\[\text{discusses}\]

\[\text{discusses}\]

\[\text{discusses}\]

\[\text{discusses}\]

\[\text{discusses}\]

30 min ago 20 min ago 10 min ago now

http://streamreasoning.org/sr4ld2013
Not so naïve data-driven stream reasoning

- Memo

The inference process is performed incrementally

Reasoner

Inferred data

S2R

insertions

deletions

SPARQL

ontology

Incremental !!!

The inference process is performed incrementally

p₁ → p₂

p₂ ↔ p₄

p₁ ↔ p₃

p₃ ↔ p₅

p₄ ↔ p₅

p₅ ↔ p₆

40 min ago | 30 min ago | 20 min ago | 10 min ago | now | now

http://streamreasoning.org/sr4ld2013
Not so naïve data-driven stream reasoning

- Memo

The inference process is performed incrementally

The diagram shows a process involving reasoner S2R, ontology, and SPARQL, with incremental updates for insertions and deletions.
Not so naïve data-driven stream reasoning

- Memo

```
S2R

insertions

Reasoner

Inferred data

ontology

SPARQL

```

Incremental！！！

No inference is required when nothing changes
Not so naïve data-driven stream reasoning

**Memo**

- S2R
- Inference is required to process deletions :-(
- Reasoner
- Inferred data
- Incremental !!!
- SPARQL

Inference is required to processes deletions :-(
Wrapping up Naïve Stream Reasoning

- Data-driven is very expensive
  - It materializes the full RDF snapshot delivered by the S2R operator
  - It infers data even if it is not relevant to the Q/A task

- Query-driven (backward) is expensive
  - It infers only data relevant to the Q/A task
  - It does not benefit from data inferred for the previous snapshot delivered by the S2R operator

- Query-driven by query rewriting is problematic
  - Lack of a standard query language and well-understood operational semantics for DSMS and CEP
  - Lack of expressiveness in OWL2QL

- Not so naïve stream reasoning using DRed is practicable
  - The inference process is performed incrementally
  - No inference is required when nothing changes
  - Inference is required to process deletions :-(

http://streamreasoning.org/sr4ld2013
Naive reasoning on RDF streams

Emanuele Della Valle
emanuele.dellavalle@polimi.it
http://emanueledellavalle.org