Stream Reasoning: mastering the velocity and the variety dimensions of Big Data at once

Emanuele Della Valle
DEIB - Politecnico di Milano
@manudellavalle
emanuele.dellavalle@polimi.it
http://emanueledellavalle.org
It's a streaming world ...

- Off-shore oil operations
- Smart Cities
- Global Contact Center
- Social networks
- Generate data streams!

... looking for reactive answers ...

- What is the expected time to failure when that turbine's barring starts to vibrate as detected in the last 10 minutes?

- Is public transportation where the people are?

- Who are the best available agents to route all these unexpected contacts about the tariff plan launched yesterday?

- Who is driving the discussion about the top 10 emerging topics?

- Require continuous processing and reactive answer
... and many more conflicting requirements

A system able to answer those queries must be able to

• handle **massive datasets**
• process **data streams** on the fly
• cope with **heterogeneous datasets**
• cope with **incomplete data**
• cope with **noisy data**
• provide **reactive answers**
• support **fine-grained access**
• integrate **complex domain models**

In **Big Data** terms

<table>
<thead>
<tr>
<th></th>
<th>Volume</th>
<th>Velocity</th>
<th>Variety</th>
<th>Veracity</th>
</tr>
</thead>
<tbody>
<tr>
<td>SR 2015, Austria 3.11.2015</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>@manudellavalle</td>
<td><a href="http://emanueledellavalle.org">http://emanueledellavalle.org</a></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Grand challenge

- Volume + Velocity + Variety = hard deal
A good reason to embrace it!

• ++ Variety $\rightarrow$ ++ value
From challenges to opportunities

• Formally data streams are:
  – unbounded sequences of time-varying data elements

  ![Diagram of unbounded data stream](image)

  \( \text{time} \)

• Less formally, in many application domains, they are:
  – a “continuous” flow of information
  – where recent information is more relevant as it describes the current state of a dynamic system

• Opportunities
  – Forget old enough information
  – Exploit the implicit ordering (by recency) in the data
State-of-the-art: DSMS and CEP

- A paradigmatic change!
- Continuous queries registered over streams that are observed through windows

Dynamic System

Registered Continuous Query

input streams

window

streams of answer
## DSMS and CEP vs. requirements

<table>
<thead>
<tr>
<th>Requirement</th>
<th>DSMS</th>
<th>CEP</th>
</tr>
</thead>
<tbody>
<tr>
<td>massive datasets</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>data streams</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>heterogeneous dataset</td>
<td></td>
<td>✗</td>
</tr>
<tr>
<td>incomplete data</td>
<td></td>
<td>✗</td>
</tr>
<tr>
<td>noisy data</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>reactive answers</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>fine-grained information access</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>complex domain models</td>
<td></td>
<td>✗</td>
</tr>
</tbody>
</table>
# DSMS/CEP, OBDA vs. requirements

<table>
<thead>
<tr>
<th>Requirement</th>
<th>DSMS CEP</th>
<th>OBDA</th>
</tr>
</thead>
<tbody>
<tr>
<td>massive datasets</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>data streams</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>heterogeneous dataset</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>incomplete data</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>noisy data</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>reactive answers</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>fine-grained information access</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>complex domain models</td>
<td>✗</td>
<td>✓</td>
</tr>
</tbody>
</table>
Stream Reasoning

• Research question
  – is it possible to make sense in real time of multiple, heterogeneous, gigantic and inevitably noisy and incomplete data streams in order to support the decision processes of extremely large numbers of concurrent users?

• Proposed approach

Sub-research questions

1. Is it possible **extend the Semantic Web stack** in order to represent heterogeneous data streams, continuous queries, and continuous reasoning tasks?

2. Does the ordered nature of data streams and the possibility to forget old enough information allow to **optimize continuous querying and continuous reasoning tasks so to provide reactive answers** to large number of concurrent users without forsaking correctness or completeness?

3. Can Semantic Web and Machine Learning technologies be jointly employed to **cope with the noisy and incomplete nature of data streams**?

4. Are there **practical cases** where processing data stream at semantic level is the best choice?
Sub-research questions

1. Is it possible extend the Semantic Web stack in order to represent heterogeneous data streams, continuous queries, and continuous reasoning tasks?

2. Does the ordered nature of data streams and the possibility to forget old enough information allow to optimize continuous querying and continuous reasoning tasks so to provide reactive answers to large number of concurrent users without forsaking correctness or completeness?

3. Can Semantic Web and Machine Learning technologies be jointly employed to cope with the noisy and incomplete nature of data streams?

4. Are there practical cases where processing data stream at semantic level is the best choice?
Contribution: RDF stream Models

- **RDF Stream** (the C-SPARQL way)
  - Unbound sequence of time-varying triples
  - each represented by a pair made of an RDF triple and its timestamp
  - Timestamp are non-decreasing (allowing for simultaneity)

```
... @BarakObama posts "Four more years", 8:16PM 6 Nov 2012
@Alice posts "RT: Four more years", 8:17PM 6 Nov 2012
...
```

Contribution: RDF stream Models

• **RDF Stream** *(the Streaming Linked Data way)*
  – Unbound sequence of time-varying graphs
  – each represented by a pair made of an RDF graph and its timestamp
  – Timestamps (if present) are monotonically increasing
  – Graphs act as a form of punctuation (all triples in a graph are simultaneous)

Work in progress

• In 2013, an RDF Stream Processing (RSP) community group was created at W3C
  
  http://www.w3.org/community/rsp/

• RSP data model and serialization
  
  – https://github.com/streamreasoning/RSP-QL/blob/master/Serialization.md
Contribution: Continuous-SPARQL
Contribution: Continuous-SPARQL

Who are the opinion makers? i.e., the users who are likely to influence the behavior their followers

REGISTER STREAM OpinionMakers COMPUTED EVERY 5m AS
CONSTRUCT { ?opinionMaker sd:about ?resource } FROM STREAM <http://...> [RANGE 30m STEP 5m]
WHERE {
}
HAVING ( COUNT(DISTINCT ?follower) > 3 )
**Contribution: Continuous-SPARQL**

Who are the opinion makers? i.e., the users who are likely to influence their followers' behavior.

```
```

Alternatives to C-SPARQL

• CQELS
  – What: STREAM clause, focus on new answer

• SPARQL\textsubscript{Stream}
  – What: window in the past, focus on RDF to Stream operators

• EP-SPARQL
  – What: focus on event specific operators

• TEF-SPARQL
  – What: adds "facts" as first class elements
  – Ref: https://www.merlin.uzh.ch/publication/show/8467
## Alternatives to C-SPARQL

### Comparison between existing approaches

<table>
<thead>
<tr>
<th>System</th>
<th>S2R</th>
<th>R2R</th>
<th>Time-aware</th>
<th>R2S</th>
</tr>
</thead>
<tbody>
<tr>
<td>C-SPARQL Engine</td>
<td>Logical and triple-based</td>
<td>SPARQL 1.1 query</td>
<td>timestamp function</td>
<td>Batch only</td>
</tr>
<tr>
<td>Streaming Linked Data Framework</td>
<td>Logical and graph-based</td>
<td>SPARQL 1.1</td>
<td>no</td>
<td>Batch only</td>
</tr>
<tr>
<td>SPARQL\textsubscript{stream}</td>
<td>Logical and triple-based</td>
<td>SPARQL 1.1</td>
<td>no</td>
<td>Ins, batch, del</td>
</tr>
<tr>
<td>CQELS</td>
<td>Logical and triple-based</td>
<td>SPARQL 1.1</td>
<td>no</td>
<td>Ins only</td>
</tr>
<tr>
<td>TEF-SPARQL</td>
<td>no</td>
<td>SPARQL-like</td>
<td>Temporarily Facts, BEFORE SINCE, UNTIL, DURING,</td>
<td>Batch only</td>
</tr>
<tr>
<td>EP-SPARQL</td>
<td>no</td>
<td>SPARQL 1.0</td>
<td>SEQ, PAR, AND, OR, DURING, STARTS,_EQUALS, NOT, MEETS, FINISHES</td>
<td>Ins only</td>
</tr>
</tbody>
</table>
Work in progress at RSP@W3C

- **RSP-QL**
  - Syntax
  - Proposed semantics
  - Semantics (work in progress)
  - Quick ref.
Contribution: continuous deductive reasoning

• DL Ontology Stream $S^T$
  – A ontology stream with respect to a static Tbox $T$ is a sequence of Abox axioms $S^T(i)$

• A Windowed Ontology Stream $S^T(o,c]$
  – A windowed ontology stream with respect to a static Tbox $T$ is the union of the Abox axioms $S^T(i)$ where $0<i \leq c$

• Reasoning on a Windowed Ontology Stream $S^T(o,c]$ is as reasoning on a static DL KB

Emanuele Della Valle, Stefano Ceri, Davide Francesco Barbieri, Daniele Braga, Alessandro Campi: **A First Step Towards Stream Reasoning.** FIS 2008: 72-81
Example of continuous deductive reasoning

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 ]
}
Finding

• The Semantic Web stack can be extended so to incorporate streaming data as a first class citizen
  – RDF stream data model
  – Continuous SPARQL syntax and semantics
  – Continuous deductive reasoning semantics
Alternatives to continuous deductive (RDFS++) reasoning

- **ETALIS**
  - What: RDFS + Allen Algebra

- **STARQL**
  - What:
    - DL-Lite + Conjunctive Query + time-series
    - SHI + Grounded Conjunctive Queries + time-series

- **ASP-based**
  - What: time-decaying ASP

- **LARS**
  - What: high-level unified formal foundation for stream reasoning
Sub-research questions

1. Is it possible to extend the Semantic Web stack in order to represent heterogeneous data streams, continuous queries, and continuous reasoning tasks?

2. Does the ordered nature of data streams and the possibility to forget old enough information allow to optimize continuous querying and continuous reasoning tasks so to provide reactive answers to large number of concurrent users without forsaking correctness or completeness?

3. Can Semantic Web and Machine Learning technologies be jointly employed to cope with the noisy and incomplete nature of data streams?

4. Are there practical cases where processing data stream at semantic level is the best choice?
Contribution: optimize querying for reactive answers

- **C-SPARQL engine** time window-based selection outperforms SPARQL filter-based selection (Jena-ARQ)

Our In-memory RDF stream processing engine

---

D. Barbieri, D. Braga, S. Ceri, E. Della Valle, Y. Huang, V. Tresp, A. Rettinger, H. Wermser:
*Deductive and Inductive Stream Reasoning for Semantic Social Media Analytics*
Not so naïve approach to stream reasoning

• 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

Is DRed needed?

• **DRed** works with **random insertions and deletions**
• **In a streaming setting**, when a triple enters the window, given the size of the window, the reasoner knows already when it will be deleted!

  • E.g.,
    ‒ if the window is 40 minutes long, and,
    ‒ it is 10:00, the triple(s) entering now
    ‒ will exit on 10:40.

• **Conclusion**
  ‒ deletions are predictable

<table>
<thead>
<tr>
<th>Time</th>
<th>Enter window</th>
<th>Exit window</th>
<th>Explicitly in window</th>
</tr>
</thead>
<tbody>
<tr>
<td>10:00</td>
<td>A←B</td>
<td>A ← B</td>
<td></td>
</tr>
<tr>
<td>10:10</td>
<td>B←C</td>
<td>A ← B←C</td>
<td></td>
</tr>
<tr>
<td>10:20</td>
<td>A←E</td>
<td>A ← B←C</td>
<td>E</td>
</tr>
<tr>
<td>10:30</td>
<td>E←C</td>
<td>A ← B←C</td>
<td>E</td>
</tr>
<tr>
<td>10:40</td>
<td>A←B</td>
<td>A ← B←C</td>
<td>E</td>
</tr>
<tr>
<td>10:50</td>
<td>B←C</td>
<td>A ← B←C</td>
<td>E</td>
</tr>
<tr>
<td>11:00</td>
<td>A←E</td>
<td>A ← B←C</td>
<td>E</td>
</tr>
</tbody>
</table>
Contribution: IMaRS algorithm

• Idea:
  – add an expiration time to each triple and
  – use an hash table to index triples by their expiration time

• The algorithm
  1. deletes expired triples
  2. Adds the new derivations that are consequences of insertions annotating each inferred triple with an expiration time (the min of those of the triple it is derived from), and
  3. when multiple derivations occur, for each multiple derivation, it keeps the max expiration time.
Contribution: IMaRS algorithm

- Incremental Reasoning on RDF streams (IMaRS): new reasoning algorithm optimized for reactive query answering

  - Re-materialize after each window slide
  - Use DRed
  - IMaRS

---


Contribution: IMaRS algorithm

- comparison of the average time needed to answer a C-SPARQL query, when 2% of the content exits the window each time it slides, using
  - A backward reasoner on the window content
  - DRed + standard SPARQL on the materialization
  - IMaRS + standard SPARQL on the materialization

<table>
<thead>
<tr>
<th></th>
<th>Backward Reasoner</th>
<th>DRed + SPARQL</th>
<th>IMaRS + SPARQL</th>
</tr>
</thead>
<tbody>
<tr>
<td>query</td>
<td>582</td>
<td>161</td>
<td>161</td>
</tr>
<tr>
<td>materialization</td>
<td>0</td>
<td>1591</td>
<td>28</td>
</tr>
</tbody>
</table>

SR 2015, Austria - 3.11.2015

@manudellavalle - http://emanueledellavalle.org
Finding

• **Stream Reasoning task is feasible** and the very nature of streaming data offers opportunities to optimise reasoning tasks where data is ordered by recency and can be forgotten after a while
  
  – **C-SPARQL Engine prototype**
  
  – **IMaRS** continuous incremental reasoning algorithm
Optimizing for stream reasoning alternative approaches

• DyKnow
  – How: logical models of an observed dynamic system + metric temporal logics

• MorphStream
  – How: rewriting in DSMS languages (one at a time)

• TR-Owl
  – How: Truth maintenance for EL++ with syntactic approximations

• ETALIS
  – How: rewriting in prolog

(continues in the next slide)
Optimizing for stream reasoning alternative approaches

- **Sparkwave**
  - How: extended RETE algorithm for windows and RDFS

- **DynamiTE**
  - How: Truth maintenance for ρDF (a fragment of RDFS)

- **STARQL**
  - How: rewriting on a scalable DSMS with time-series support

- **ASP-based**
  - How: optimizing ASP for incremental and time-decaying programs

- **The Backward/Forward Algorithm**
  - How: optimizing DRed
Sub-research questions

1. Is it possible extend the Semantic Web stack in order to represent heterogeneous data streams, continuous queries, and continuous reasoning tasks?

2. Does the ordered nature of data streams and the possibility to forget old enough information allow to optimize continuous querying and continuous reasoning tasks so to provide reactive answers to large number of concurrent users without forsaking correctness or completeness?

3. Can Semantic Web and Machine Learning technologies be jointly employed to cope with the noisy and incomplete nature of data streams?

4. Are there practical cases where processing data stream at semantic level is the best choice?
Cope with the noisy and incomplete data

- "Noise" is reduced using **DSMS** techniques
- **Deductive stream reasoning** copes with **incompleteness** deducing implicit facts
- **Inductive stream reasoning** copes with "irrepairable" incompleteness inducing missing facts

---

Findings

• A combination of deductive and inductive stream reasoning techniques can cope with incomplete and noisy data
Alternative approaches

• Stream Reasoning with Probabilistic Answer Set Programming
  – Anni-Yasmin Turhan, Erik Zenker: Towards Temporal Fuzzy Query Answering on Stream-based Data. HiDeSt@KI 2015: 56-69
Sub-research questions

1. Is it possible to extend the Semantic Web stack in order to represent heterogeneous data streams, continuous queries, and continuous reasoning tasks?

2. Does the ordered nature of data streams and the possibility to forget old enough information allow to optimize continuous querying and continuous reasoning tasks so to provide reactive answers to large number of concurrent users without forsaking correctness or completeness?

3. Can Semantic Web and Machine Learning technologies be jointly employed to cope with the noisy and incomplete nature of data streams?

4. Are there practical cases where processing data stream at semantic level is the best choice?
Contribution:
Streaming Linked Data Framework

Contribution: RSP services

- RSP services: a RESTful interface for RSP engines

- [http://streamreasoning.org/download/rsp-services](http://streamreasoning.org/download/rsp-services)
Practical cases

- **10+ deployments** in Sensor Networks & Social media analytics, e.g.

**BOTTARI**

Winner of Semantic Web Challenge 2011

**City Data Fusion**

Winner of IBM faculty award 2013

**Social Listener**

M. Balduini, I. Celino, D. Dell’Aglio, E. Della Valle, Y. Huang, T. Lee, S.-H. Kim, V. Tresp:

**BOTTARI**: An augmented reality mobile application to deliver personalized and location-based recommendations by continuous analysis of social media streams. J. Web Sem. 16: 33-41 (2012)

M. Balduini, E. Della Valle, M. Azzi, R. Larcher, F. Antonelli, and P. Ciuccarelli:


@manudellavalle - http://emanueledellavalle.org
Findings

1. **The Semantic Web stack can be extended** so to incorporate streaming data as a first class citizen
   - **RDF stream** data model
   - **Continuous SPARQL** syntax and semantics
   - **Continuous deductive reasoning** semantics

2. **Stream Reasoning task is feasible** and the very nature of streaming data offers opportunities to **optimise reasoning tasks** where data is ordered by recency and can be forgotten after a while
   - **IMaRS** continuous incremental reasoning **algorithm**
   - **C-SPARQL** Engine prototype

3. A combination of **deductive and inductive stream reasoning techniques** can **cope with incomplete and noisy data**

4. There are **application domains** where Stream Reasoning offers an adequate solution
Open issues

1. The Semantic Web stack can be extended
   – "Navigating the Chasm between the Scylla of Practical Applications and the Charybdis of Theoretical Approaches"
   A. Bernstein, 2015

2. Stream Reasoning task is feasible
   – It's time to start removing assumptions
     • knowledge does not change
     • background data does not change
   – OBDA for SQL ≠ OBDA for continuous querying

3. Stream reasoning can cope with incomplete and noisy data
   – Theory is needed!

4. There are application domains where Stream Reasoning offers an adequate solution
   – Rigorous quantitative comparative research is needed
Advertisements :-P

- Check out my PhD thesis
  - [http://dare.ubvu.vu.nl/handle/1871/53293](http://dare.ubvu.vu.nl/handle/1871/53293)
  - Chapter 1: Introduction
    - The content of this presentation
  - Chapter 8: conclusions
    - A review of stream reasoning approaches updated in spring 2015

- Put an "I like" to Stream Reasoning on Facebook
  - [https://www.facebook.com/streamreasoning](https://www.facebook.com/streamreasoning)
Stream Reasoning: mastering the velocity and the variety dimensions of Big Data at once

Emanuele Della Valle
DEIB - Politecnico di Milano
@manudellavalle
emanuele.dellavalle@polimi.it
http://emanueledellavalle.org