Taming Velocity and Variety Simultaneously in Big Data with Stream Reasoning

Emanuele Della Valle, Daniele Dell’Aglio, Alessandro Margara
DEIB - Politecnico di Milano
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 a credits slide stating

These slides are partially based on “Tutorial on Stream Reasoning: Managing Velocity and Variety in Big Data at DEBS 2016” by E. Della Valle, D. Dell’Aglio and A. Margara

[http://streamreasoning.org/events/srdebs2016](http://streamreasoning.org/events/srdebs2016)

To view a copy of this license, visit

[http://creativecommons.org/licenses/by/3.0/](http://creativecommons.org/licenses/by/3.0/)
Big Data

DEBS

Semantic Web

Volume

Velocity

Real time
Near real time
Micro-batch
Batch

Database

Documents
Web Images
Social media
Unstructured

Della Valle, Dell’Aglio, Margara
Big Data

Velocity

Real time
Near real time
Micro-batch
Batch

Stream Reasoning

Variety

Database
Documents
Images
Web
Social media
Unstructured

Volume

MB
GB
TB
PB

Della Valle, Dell’Aglio, Margara
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?

H. Stuckenschmidt, S. Ceri, E. Della Valle, F. van Harmelen
Towards Expressive Stream Reasoning.
• Introduction to the Semantic Web
  • Data model
  • Query model
  • Reasoning

• Stream Reasoning
  • Data model
  • Query model
  • Naïve reasoning

• Stream Reasoning
  • State-of-the-art approaches
  • Discussion
  • Conclusions
The Web today

Large number of integrations
• ad hoc
• pair-wise

Too much information to browse, need for searching and mashing up automatically

Each site is “understandable” for us
Computers don’t “understand” much
The semantic gap problem

Symbolic description

Semantic gap

Sensor data
Understanding means bridging the gap

Symbolic description

Understanding

Sensor data
Understanding: smart machines

Symbolic description

Working examples found on the Web
Image processing
retrievr: find by sketching
http://labs.systemone.at/retrievr/

Audio processing
midomi: find by singing
http://www.midomi.com/

Natural Language Processing
semantic proxy:
http://semanticproxy.opencalais.com/about.html
Smart machines alone cannot bridge the gap ...

Natural language processing (NLP) meets image processing (IP)

*NLP*: What does your eye see?

*IP*: I see a sea

*NLP*: You see a “c”?

*IP*: Yes, what else could it be?

[Source NLP Related Entertainment](http://www.cl.cam.ac.uk/Research/NL/amusement.html)
Della Valle, Dell’Aglio, Margara

... smart data is also needed

Natural language processing (NLP) meets image processing (IP)

NLP: What does your eye see?
IP: I see a wordnet:word-sea
NLP: mmh, I see a wordnet:word-c
IP: I believe we have different understanding of the world...
NLP: So do I

The Semantic Web offers a set of standards to enable smart data at Web scale
What a machine understands of the Web

What we say to Web agents

• "For more information visit <a href="http://www.ex.org">my company</a> Web site ...”

What do they “hear”?  

• "blah blah blah blah blah <a href="http://www.ex.org">blah blah blah</a> blah blah ...”

• Yet, this is enough for them to perform some tasks for us
What does Google understand?

- From: [page1] links [page2] ...
- ... Google understands that page2 is interesting

Google is able to rank results!

“PageRank™ […] relies on the uniquely democratic nature of the web by using its vast link structure as an indicator of an individual page's value.”

http://www.google.com/technology/
The Semantic Web

“The Semantic Web is not a separate Web, but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation.”


Key concepts

• **an extension** of the current Web
• in which **information is given well-defined meaning**
• better enabling **computers** and **people** to work in cooperation
  • Both for computers and people
The Semantic Web: An extension to the current Web

Web 1.0

The Web Today
The Semantic Web: Information is given a well defined meaning

Web 1.0

Human understandable but "only" machine-readable

Semantic Web

Human and machine "understandable"
The Semantic Web: Enables computers and people to better work in cooperation

Fewer ad-hoc integrations
- *Standard*
- *Multi-lateral*

Easier to understand for people
More “understandable” for computers
Linking Open Data Project

Goal: extend the Web by publishing open data sets using Semantic Web technologies

Visit [http://esw.w3.org/topic/SweolG/TaskForces/CommunityProjects/LinkingOpenData](http://esw.w3.org/topic/SweolG/TaskForces/CommunityProjects/LinkingOpenData)!
Example: BIO2RDF

Peter Ansell, Model and prototype for querying multiple linked scientific datasets
Example: LinkedGeoData

- Tries to add a spatial dimension to the Semantic Web.
- Uses the information collected by the OpenStreetMap project
- Makes it available as Linked Data
- Interlinks with other knowledge bases in the Linking Open Data initiative.
European Union Open Data Portal

Find datasets...

Show results with:  all of these words |  any of these words |  the exact phrase

Total datasets available: 7846

Most viewed datasets
- DGT-Translation Memory (13045 views)
- Elevation map of Europe (9886 views)
- CORDIS – EU research projects under Horizon 2020

Browse datasets by subject
- Employment and working conditions
- Social questions
- Economics
- Finance

Suggest a dataset
Is there a dataset from the EU that you could not find in this portal?

Please request the dataset>>

Della Valle, Dell’Aglio, Margara
Example: Best Buy

Nikon - 12.3-Megapixel Digital SLR Camera - Black

Model: D90 Body Only | SKU: 9856675

4.8 of 5 (4 reviews)

<div rel="v:hasReview">
  <span property="v:rating" datatype="xsd:string">4.8</span>
of
  <span property="v:best">5</span>
</div>
Example: Best Buy

Google for Nikon+12.3-Megapixel+Digital+SLR+Camera

Nikon-d90 - Super offerte - Grandi sconti per Nikon-d90
www.nextag.it/Computer
Scopri le nostre migliori offerte!

Nikon D90 | Kelkoo.it
www.kelkoo.it/Fotocamere
Prezzi Nikon D90 Confronta i Prezzi e Risparmia!

Amazon.com: Nikon D300 DX 12.3MP Digital SLR Camera with 18 ...
www.amazon.com › ... › Digital Cameras › Digital SLR Cameras
Engineered with pro-level features and performance, the 12.3-effective-megapixel D300 combines brand new technologies with advanced features inherited ...

Shopping results for Nikon 12.3-Megapixel Digital SLR Camera

Nikon D90 Digital SLR Camera with Nikon AF-S DX 18 ...
⭐⭐⭐⭐⭐ 947 reviews - $852 - 50 stores
Nikon D700 Digital SLR Camera (Body Only)
⭐⭐⭐⭐⭐ 1,147 reviews - $2,249 - 36 stores
Nikon D90 Digital SLR Camera (Body Only)
⭐⭐⭐⭐⭐ 1,312 reviews - $550 - 41 stores
Use RDFa with some FB specific vocabulary

- **og:title** - The title of your object, e.g., "The Rock"
- **og:type** - The type of your object, e.g., "movie"
- **og:image** - An image URL
- **og:url** - The permanent ID of your object
- **og:description** - A one to two sentence description of your object
- **og:site_name** - If your object is part of a larger web site, the name which should be displayed for the overall site. e.g., "IMDb"

http://ogp.me/
Example: Google structured data

Enabling Rich Snippets for Events

Event markup describes the details of organized events, such as musical concerts or art festivals, that people may attend at a particular time and place. Events that meet Google’s policy guidelines may be eligible for Rich Snippets in search results, as described in this document, as well as for display in Knowledge-Graph-powered features, documented separately.

Event Rich Snippets add one or more lines with structured details of upcoming events below your search result, like this:

[source https://developers.google.com/structured-data/rich-snippets/]
• Initiative launched on June 2011 by Bing, Google and Yahoo!
  • To “create and support a common set of schemas for structured data markup on web pages”
• Broad type hierarchy
  • See http://schema.org/docs/full.html
• Recommends either microdata or RDFa
Semantic Web layers

- **Already Possible**
  - User Interface & Applications
  - Trust
  - Proof

- **Under Investigation**
  - Unifying Logic

- **Standardized**
  - Query: SPARQL
  - Ontology: OWL
  - Rule: RIF
  - RDFS
  - Data interchange: RDF
  - XML
  - URI/IRI

[source](http://www.w3.org/2007/03/layerCake.png)
Data Interchange: RDF

User Interface & Applications

Trust

Proof

Unifying Logic

Query: SPARQL

Ontology: OWL

Rule: RIF

Crypto

Data interchange: RDF

XML

URI/IRI
Looking for a flexible data model

• Why
  • Applications change frequently
    • New features
  • Graceful evolution is important

• Optimal: relational model
  • Relational model is remarkably flexible
  • Supports graceful evolution
    • Easily accommodates new data
      • Add another table
    • Existing queries are unaffected
  • Allows data to be easily combined ("joined")
  • 25+ years of relational database experience
The adaptation of the relational model to the Web give rise to RDF
From *tuples* to *triples*

Any relational data can be represented as triples
Row Key --> Subject
Column --> Property
Value --> Value
Representing relational data in RDF

E.g., drug data

<table>
<thead>
<tr>
<th>ID</th>
<th>Category</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>BB.2</td>
<td>Anxiolytics</td>
<td>(\text{C}<em>{16}\text{H}</em>{21}\text{NO}_2)</td>
</tr>
</tbody>
</table>

Represented in RDF

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Lang</th>
</tr>
</thead>
<tbody>
<tr>
<td>BB.2</td>
<td>Propranolol</td>
<td>en</td>
</tr>
<tr>
<td>BB.2</td>
<td>Propranololo</td>
<td>it</td>
</tr>
<tr>
<td>BB.2</td>
<td>プロプラノロール</td>
<td>jp</td>
</tr>
</tbody>
</table>
Representing data in RDF
Where to find common URIs

• 1st solution: you know them ;-)
  • RDF vocabulary already includes:
    • rdf:type: the “instance of” relationship between an individual and its class
    • rdf:Property: the “concept” of property
  • RDF comes along a schema description language (RDF-S)
    • rdfs:Class: the “concept” of class
    • rdfs:subClassOf: the “is a” relationship between two classes
    • rdfs:label: the standard way to provide a human-readable label

• 2nd you search for them
  • Specialized search engines (http://vocab.cc/)
  • Repositories (http://lov.okfn.org/dataset/lov/)
Query Language

Query: SPARQL
• RDF connects entities in a graph

• SPARQL queries such graph
• Idea: pattern matching on the RDF graph

• Triple patterns are triples where each part can be replaced with a variable

\( ?s \ a \ \text{dbpedia-owl:Drug} \)

• Matching:
  • find a set of bindings for \(?s\)
  • such that the substitution of variables with values creates triples that are in the dataset
A basic graph pattern is a set of triple patterns, all of which must be matched.

E.g.,

\[
\text{?s a dbpedia-owl:Drug} \\
\text{?s skos:subject category:Anxiolytics} \\
\text{?s ?p dbpedia:Kidney}
\]

Further operators extend the complexity of the pattern ...
Complex graph patterns

- RDF is "semi structured" and has no integrity constrains
- SPARQL addresses this issue with:

  - **Group** patterns match if all sub-patterns match
    - Syntax: \{ ... \}

  - **OPTIONAL** graph patterns add information to a result if it is available, but do not stop the query if the information is missing
    - Syntax: OPTIONAL \{ ... \}

  - **UNION** graph patterns match alternatives
    - Syntax: \{ ... \} UNION \{ ... \}
SPARQL

- SELECT
- CONSTRUCT
- ASK
- DESCRIBE

• Ordered by
• Limit N
• Offset N
• Distinct
• Aggregate
• Project
SPARQL

- Allows data translation from one vocabulary to another one
SPARQL

**Query Form**

- **SELECT**
- **CONSTRUCT**
- **ASK**
- **DESCRIBE**

- Returns true if the query has results
- Used to check consistency
**SPARQL**

- **Query Form**
  - SELECT
  - CONSTRUCT
  - ASK
  - DESCRIBE

- **TRUE/FALSE**

- Returns a server-side description of the required RDF resource
Ontology: RDF-S and OWL
What is an ontology?

• Philosophy, 400BC
  • Systematic explanation of Existence

• Gruber, 1993
  • Explicit specification of a conceptualization

• Borst, 1997
  • Formal specification of a shared conceptualization

• Studer, 1998
  • Formal, explicit specification of a shared conceptualization
What is an ontology?

Formal, explicit specification of a shared conceptualization

- Machine readable
  - Makes domain assumptions explicit
  - Several people agree that the conceptual model is adequate to describe the domain of interest
- A conceptual model of some aspects of the reality
What is an ontology?

- A model of (some aspect of) the world
- Introduces a **vocabulary**
- Specifies **meaning** (semantics) of terms
  - **Heart** is a muscular **organ** that is **part of** the **circulatory system**
- Is **formalized** using a suitable logic
  - $\forall x. ([\text{Heart}(x) \rightarrow \text{MuscularOrgan}(x) \land \exists y. [\text{isPartOf}(x, y) \land \text{CirculatorySystem}(y)])$
Example: a simple ontology

- Artist
  - Painter
  - Sculptor
- Piece
  - Paint
  - Sculpt

- Painter paints Piece
- Sculptor sculpts Piece
- Artist creates Piece
Specifying classes, sub-classes and instances

Creating a class
   Artist rdf:type rdfs:Class

Creating a subclass
   Painter rdfs:subClassOf Artist
   Sculptor rdfs:subClassOf Artist

Creating an instance
   Rodin rdf:type Sculptor
Creating a property

creates rdf:type rdf:Property

Using a property

Rodin creates TheKiss

Creating subproperties

paints rdfs:subPropertyOf creates
sculpts rdfs:subPropertyOf creates
Specifying domain / range constraints

Checking which classes and properties can be use together

creates rdfs:domain Artist
creates rdfs:range Piece
paints rdfs:domain Painter
paints rdfs:range Paint
sculpts rdfs:domain Sculptor
sculpts rdfs:range Sculpt
<table>
<thead>
<tr>
<th>IF</th>
<th>THEN</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x ) rdfs:subClassOf ( y ) ( a ) rdfs:subClassOf ( z )</td>
<td>( a ) rdfs:subClassOf ( x ) rdfs:subClassOf ( y ) ( a ) rdfs:subClassOf ( z )</td>
</tr>
</tbody>
</table>
A recipient, that only receives the following triple ...

\[
\text{<RDF>}
\text{<Description about="Rodin">}
\text{<sculpts resource="TheKiss"/>}
\text{</Description>}
\text{</RDF>}
\]

... can answer the following queries

- What does Rodin sculpt?
- Who does sculpt TheKiss?

... but it cannot answer

- Who is Rodin?
- What is TheKiss?
- Is there any Sculptor/Scupts?
- Is there any Artist/Piece?
RDF semantics at work

We can share the following ontology ...

```rdfs
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
@prefix ex:   <http://www.ex.org/schema#>

ex:Sculptor rdfs:subClassOf ex:Artist
ex:Painter rdfs:subClassOf ex:Artist
ex:Sculpt rdfs:subClassOf ex:Piece
ex:Painting rdfs:subClassOf ex:Piece
ex:creates rdfs:domain ex:Artist
ex:creates rdfs:range ex:Piece
ex:sculpts rdfs:subPropertyOf ex:creates
ex:sculpts rdfs:domain ex:Sculptor
ex:sculpts rdfs:range ex:Sculpt

... when transmitting the triple
  ex:Rodin ex:sculpts ex:TheKiss
```
With inference

Artist
Painter

Sculptor
Rodin

creates

paints

Piece
Paint

Sculpt
TheKiss

sculpts
The receiver can also answer the following queries

- **Who is Rodin?**
  
  ```
  PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
  PREFIX ex:   <http://www.ex.org/schema#>
  SELECT ?x
  WHERE { ex:Rodin a ?x }
  
  ?x = ex:Artist, ex:Sculptor, rdfs:Resource
  ```

- **What is TheKiss?**
  
  ```
  WHERE { ex:TheKiss a ?x }
  
  ?x = ex:Sculpt, ex:Piece, rdfs:Resource
  ```
• Is there any Sculptor?
  WHERE { ?x a ex:Sculptor}
  ?x = ex:Rodin

• Is the any Artist?
  WHERE { ?x a ex:Artist }
  ?x = ex:Rodin

• Is there any Sculpt?
  WHERE { ?x a ex:Sculpt }
  ?x = ex:TheKiss

• Is there any Piece?
  WHERE { ?x a ex:Piece }
  ?x = ex:TheKiss

• Is there any Paint?
  WHERE { ?x a ex:Paint }
  0 results

• Is there any Painter?
  WHERE { ?x a ex:Painter }
  0 results
SPARQL alone cannot answer queries that require reasoning, but ...

... a reasoner can materialized inferred data for SPARQL upfront

... or at query time
More expressive power

• RDFS is a light ontological language that enables the definition of simple vocabularies

• One might also want to express:
  
  • Cardinality constraints (max, min, exactly) for properties
    • E.g., a Polygon has 3 or more edges
  
  • Property types
    • Transitive
      • E.g., hasAncestor is a transitive property
    • Inverse
      • E.g., sculpts is the inverse property of isSculptedBy
    • Symmetric
      • E.g., isCloseTo is a symmetric property
More expressive power

- One might also want to express:
  - Restrictions for properties
    - All values of a property must be of a certain kind
      - E.g., a D.O.C. wine can be only produced by a certified winery
    - Some values of a property must be of a certain kind
      - E.g., a famous painter must have painted some famous painting
  - A class is defined as a combination of other classes
    - Union, intersection, negation, ...
      - E.g., a white wine is a wine and its color is “white”
The more an ontological language is expressive the less tractable it is.

The Web Ontology Language (OWL) comes with several profiles that offers different trade-offs between expressivity and tractability.
OWL 2

- **OWL 2 QL**
  - Useful for applications
    - That use very large volumes of data
    - Query answering is the most important task
  - Query answering complexity is the same as in RDBMSs

- **OWL 2 RL**
  - Useful for applications
    - That require scalable reasoning without sacrificing too much expressive power
    - Query answering is the most important task
  - Allows for polynomial reasoning using rule-based technologies
    - Rule-based inference might return incomplete results

- **OWL 2 EL**
  - Useful for applications
    - That use ontologies that contain a very large number of properties and/or classes
  - Maximal language for which ontology consistency, instance checking, class subsumption problems are polynomial
It’s a streaming world!

Off-shore oil operations

Smart cities

Global contact centers

Social networks

E. Della Valle, S. Ceri, F. van Harmelen, D. Fensel.
It’s a Streaming World! Reasoning upon Rapidly Changing Information.
What is the expected time to failure when that turbine's barring starts to vibrate as detected in the last 10 minutes?

Which public transportation lines are used the most between 7am and 9am?

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?
RDF stream models

- **C-SPARQL**
  - Unbound sequence of time-varying triples
  - Each RDF triple has an associated timestamp
  - Timestamps are non-decreasing

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

Querying RDF streams with C-SPARQL
RDF stream models

- Streaming Linked Data
  - Unbounded sequence of time-varying graphs
  - Each RDF graph has an associated timestamp (optional)
  - Timestamps (if present) are monotonically increasing
  - Graphs act as a form of punctuation (all triples in a graph are simultaneous)

D.F. Barbieri, E. Della Valle
A Proposal for Publishing Data Streams as Linked Data
A Position Paper, LDOW (2010)
EP-SPARQL

• EP-SPARQL
  • Unbounded sequence of triples
  • Each RDF triple has an associated interval timestamp
  • The timestamp defines the interval of time in which the triple is valid

Anicic, D., Fodor, P., Rudolph, S., Stojanovic, N. 
RDF stream models

- RDF Stream Processing (RSP) W3C community group created in 2013
  - [http://www.w3.org/community/rsp/](http://www.w3.org/community/rsp/)

- RSP data model and design principles
  - [https://github.com/streamreasoning/RSP-QL/blob/master/Serialization.md](https://github.com/streamreasoning/RSP-QL/blob/master/Serialization.md)
Continuous-SPARQL

Windows

SPARQL

Relation-to-Stream Operators
Continuous-SPARQL

Who are the opinion makers?
(Users who are likely to influence the behavior of their followers)

REGISTER STREAM OpinionMakers COMPUTED EVERY 5m AS
CONSTRUCT { ?opinionMaker sd:about ?res }
FROM STREAM <http://...> [RANGE 30m STEP 5m]
WHERE {
  FILTER (cs:timestamp(?follower ?opinion ?res) >
            cs:timestamp(?opinionMaker ?opinion ?res))
}
HAVING ( COUNT(DISTINCT ?follower) > 3 )
Stream reasoning to answer queries

Data-driven (forward reasoning)

- RDF data
- Reasoner
- Inferred data
- SPARQL
- Ontology

Query-driven (forward reasoning)

- RDF data
- Reasoner
- SPARQL
- Ontology
Example of stream reasoning

Query: measure the impact of Alice's posts

Data model

Alice posts $p_1$.

Bob posts $p_2$.

$\text{Post}\,\text{discusses}\,p_1\,\text{discusses}\,p_2\,\text{discusses}\,p_3\,\text{discusses}\,p_4\,\text{discusses}\,p_5\,\text{discusses}\,p_6\,\text{discusses}\,p_7\,\text{discusses}\,p_8$
Example of stream reasoning

- What is the impact of \(p_1\) in the last hour?
  - Let’s count the number of posts that discuss it …

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

Transitive property
Naïve data-driven stream reasoning

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

ontology

SPARQL

\[ p_1 \]

now
Naïve data-driven stream reasoning

S2R → RDF data → Reasoner → Inferred data → SPARQL

ontology

discusses

10 min ago now
Naïve data-driven stream reasoning

S2R → RDF data → Reasoner → Inferred data → SPARQL

ontology

Discusses

p₁ → p₂

p₂ → p₃

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

S2R $\rightarrow$ RDF data $\rightarrow$ Reasoner $\rightarrow$ Inferred data $\rightarrow$ SPARQL

Reasoner

ontology

S2R

RDF data

Reasoner

Inferred data

SPARQL

discusses

p_1 $\rightarrow$ p_2 $\rightarrow$ p_4

p_2

discusses

p_4

discusses

p_3

discusses

p_1

30 min ago

20 min ago

10 min ago

now

Della Valle, Dell’Aglio, Margara
The entire inference process is repeated each time the S2R operator delivers new RDF data.
Naïve data-driven stream reasoning

S2R → RDF data → Reasoner → Inferred data → SPARQL

ontology

Reasoner

p1 discusses p2 discusses p4

p2

p3

p4 discusses p5

p5

p6

p7

p8

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

Della Valle, Dell’Aglio, Margara

78
Naïve data-driven stream reasoning

S2R → RDF data → Reasoner → Inferred data

ontology

Reasoner

SPARQL

1. \( p_2 \) discusses \( p_4 \) discusses \( p_7 \)
2. \( p_3 \) discusses \( p_5 \)
3. \( p_6 \) discusses
4. \( p_5 \) discusses
5. \( p_7 \) discusses
6. \( p_8 \) discusses

50 min ago → 40 min ago → 30 min ago → 20 min ago → 10 min ago → now
The reasoner infers data that is irrelevant to query answering.
Naïve query-driven stream reasoning

S2R \rightarrow RDF data \rightarrow Reasoner \rightarrow SPARQL

Ontology

$p_1$

now
Naïve query-driven stream reasoning

S2R → RDF data → Reasoner → ontology → SPARQL

discusses

p₁

p₂

10 min ago  now
Naïve query-driven stream reasoning

S2R \rightarrow RDF data \rightarrow Reasoner \rightarrow ontology \rightarrow SPARQL

discusses

\begin{align*}
 p_1 & \quad \text{discusses} \quad p_2 \\
 p_2 & \quad \text{discusses} \quad p_3
\end{align*}

20 min ago \quad 10 \text{ min ago} \quad \textbf{now}
Naïve query-driven stream reasoning

S2R → RDF data → Reasoner → ontology → SPARQL

- \( p_1 \) discusses \( p_2 \)
- \( p_2 \) discusses \( p_4 \)
- \( p_4 \) discusses \( p_3 \)

30 min ago  20 min ago  10 min ago  now
Naïve query-driven stream reasoning

The entire inference process is repeated each time the R2S operator delivers new RDF data.
Naïve query-driven stream reasoning

1. S2R → RDF data → Reasoner → ontology → SPARQL

Reasoner

- ontology

1. p1
2. p2
3. p3
4. p4
5. p5
6. p6
7. p7
8. p8

Discusses

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

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

The backward reasoner does not even start!