Scalability in RDF Stream Processing Systems

Alejandro Llaves and Oscar Corcho

Ontology Engineering Group
Universidad Politécnica de Madrid
Madrid, Spain
allaves@fi.upm.es

May 31 2015
What we do to improve Scalability in our RDF Stream Processing System

Alejandro Llaves and Oscar Corcho

Ontology Engineering Group
Universidad Politécnica de Madrid
Madrid, Spain
allaves@fi.upm.es

May 31 2015
- Towards efficient processing of RDF data streams
- Architecture overview
- Parallelizing the pre-processing of sensor data streams
- Example of use: CSIRO's Sensor Cloud
- Discussion and future work
Towards efficient processing of RDF data streams

- **Goal:** to develop a stream processing engine capable of adapting to variable conditions, such as changing rates of input data, failure of processing nodes, or distribution of workload, while serving complex continuous queries.

- **Example of query execution parallelization** (OrdRing 2014)

Storm topology example (4 nodes)
```
SELECT ?obs.value ?sensors.location
FROM NAMED STREAM <obs> [60 SEC TO NOW]
FROM NAMED STREAM <sensors> [60 SEC TO NOW]
WHERE obs.sensorId = sensors.id ;
```
morph-streams++ architecture
Parallelizing the pre-processing of sensor data streams

Methodology
1. Transform data input into field-named tuples
2. Add semantic annotations (if needed)
3. Publish tuples to multiple channels
4. Convert tuples to RDF on (query) demand

Focus
- Storm topologies
- Environmental sensor observations
- Using Semantic Sensor Network (SSN) ontology
Example of use: CSIRO's Sensor Cloud (1/3)

Sensor Cloud

- Viticulture, water management, weather monitoring, oyster farming...
- RESTful API – JSON
- Network → Platform → Sensor → Phenomenon → Observation
- Lack of semantic descriptions, e.g. rain_trace vs Rain.
- Multiple HTTP requests to query various streams.

Source: CSIRO
Example of use: CSIRO's Sensor Cloud (2/3)

1. Sensor Cloud messages to field-named tuples

   <sample time="2015-05-28T16:30" value="48" sensor="bom_gov.au.94961.air.rel_hum"/>

   sampling time

   
   system time

   
   
   phenomenon

   network

   platform

   sensor

   [-43.3167", "147.0075"]

   latitude

   longitude

2. SWEET annotations for phenomena

<table>
<thead>
<tr>
<th>Sensor Cloud phenomena</th>
<th>SWEET annotations</th>
</tr>
</thead>
<tbody>
<tr>
<td>rain_trace, Rain, rainfall-per-hour</td>
<td>Rainfall</td>
</tr>
<tr>
<td>air_temp, temperature_deg_c, temperature,</td>
<td>Temperature</td>
</tr>
<tr>
<td>average-air-temperature</td>
<td></td>
</tr>
<tr>
<td>wind_dir, average-wind-direction</td>
<td>Direction</td>
</tr>
<tr>
<td>wind_spd_kmh, average-wind-speed</td>
<td>WindSpeed</td>
</tr>
<tr>
<td>rel_hum, average-relative-humidity</td>
<td>RelativeHumidity</td>
</tr>
<tr>
<td>dewpt</td>
<td>DewPoint</td>
</tr>
<tr>
<td>Evap</td>
<td>Evaporation</td>
</tr>
</tbody>
</table>
3. Publish to multiple channels

- SC2Tuple
- SWEET Annotation
- Publish2Kafka

4. Convert tuples to SSN model on (query) demand

- All tuples
- sweet:RelativeHumidity
- sweet:Rainfall

Query: heavy rainfall events

to be continued...
Conclusion

- Division of work into simple tasks.
- Parallelize any parallelizable task.
- Delay RDF generation and convert on demand.

Future work

- Evaluation and benchmarking.
- SSN mapping interface.
- Topology package: executing distributed queries (Storm).
- theObserver (theO) package: monitoring scalability metrics for adaptive query processing.
Thanks!

The presented research has been funded by Ministerio de Economía y Competitividad (Spain) under the project "4V: Volumen, Velocidad, Variedad y Validez en la Gestión Innovadora de Datos" (TIN2013-46238-C4-2-R), by the EU Marie Curie IRSES project SemData (612551), and supported by an AWS in Education Research Grant award.

Alejandro Llaves
allaves@fi.upm.es