First Joint International Workshop on
Semantic Sensor Networks and Terra Cognita
October 11, 2015, Bethlehem, PA, USA

Emrooz: A Scalable Database for SSN Observations

Markus Stocker, Narasinha Shurpali, Kerry Taylor, George Burba, Mauno Rönkkö, Mikko Kolehmainen

markus.stocker@uef.fi
@markusstocker and @envinf

UNIVERSITY OF
EASTERN FINLAND

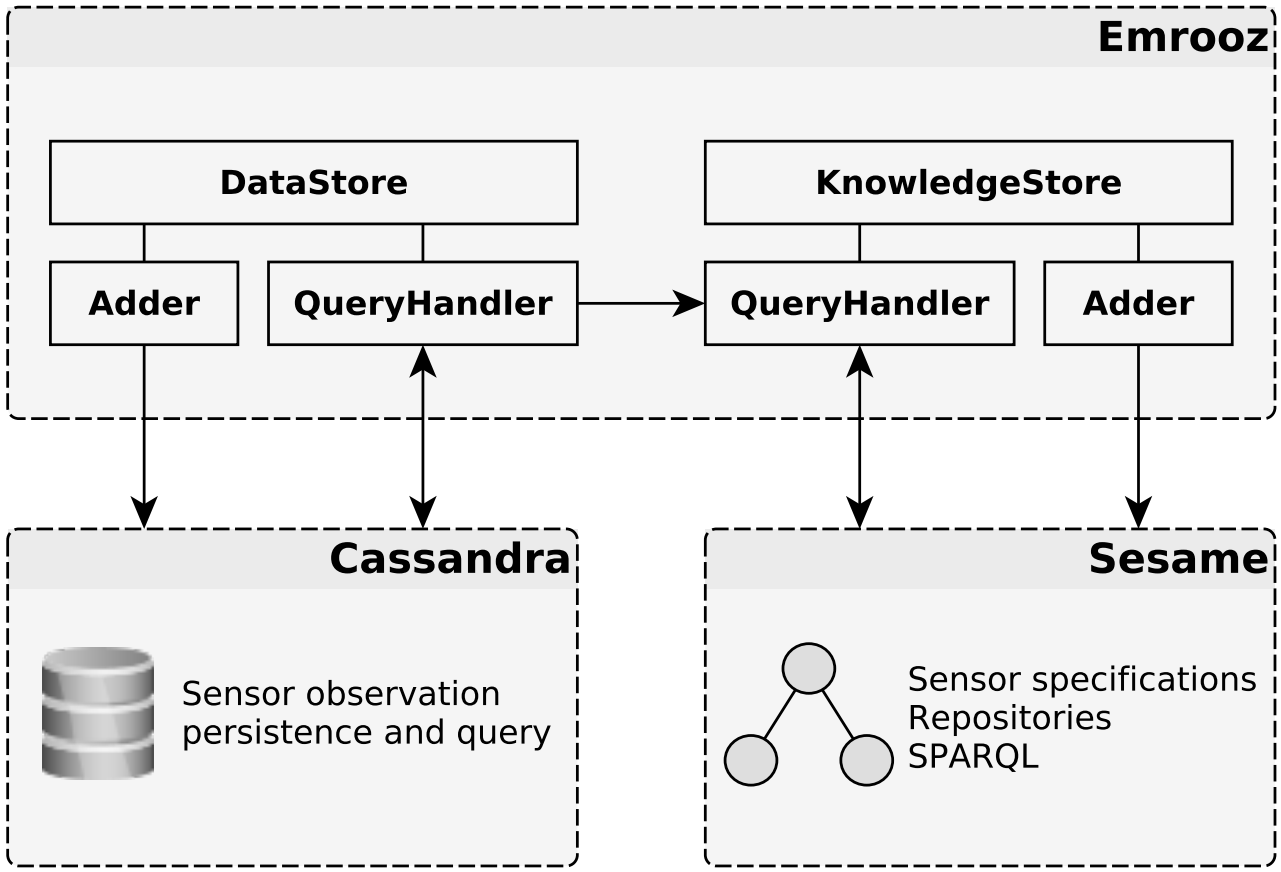
Introduction

- Expressive ontologies for sensor (meta-) data (SSN)
- Flexible graph data model (RDF)
- Triple stores obvious choice
- Unfortunately hardly viable at scale
- Triple stores indexes for graph pattern queries
- Not designed for time series interval queries

Aim

- Build a database that ...
- Consumes SSN observations in RDF
- Evaluates SSN observation SPARQL queries
- Scales to billions of observations
- Has better query performance than triple stores

Architecture



Cassandra data model

- Schema consisting of
  - Partition key (row key) of type `ascii`
  - Clustering key (column name) of type `timeuuid`
  - Column value of type `blob`
- The partition key consists of two (dash-concatenated) parts
  - SHA-256 hex string digest of sensor-property-feature URIs
  - Date time string of pattern `yyyyMMddHHmm`
    - Computed from observation result time
    - Floor-rounded to year, month, day, hour, or minute
    - Rounding depends on sensor sampling frequency
    - Goal is to limit the number of columns per row
- Clustering key determined by observation result time
- Column value is set of triples for observation (binary)

Experiments

- LI-7500A Open Path CO_2/H_2O Gas Analyzer
- LI-7700 Open Path CH_4 Analyzer
- Property of mole fraction
- Three features for the monitored gases


7200-101
Flow Module
LI-COR
LI-7550
Analyzer Interface Unit
LI-COR
LI-COR

Experiments

- January 7 to May 26, 2015, 6045 GHG archive files
- Estimated # of sensor observations is 326 430 000
- Estimated # of triples is 4.9 billion (15 triples / observation)
- Load and query performance on 10 subsets
- SPARQL query with 10 min interval
- Compared to Stardog and Blazegraph
- Test performance with varying time interval

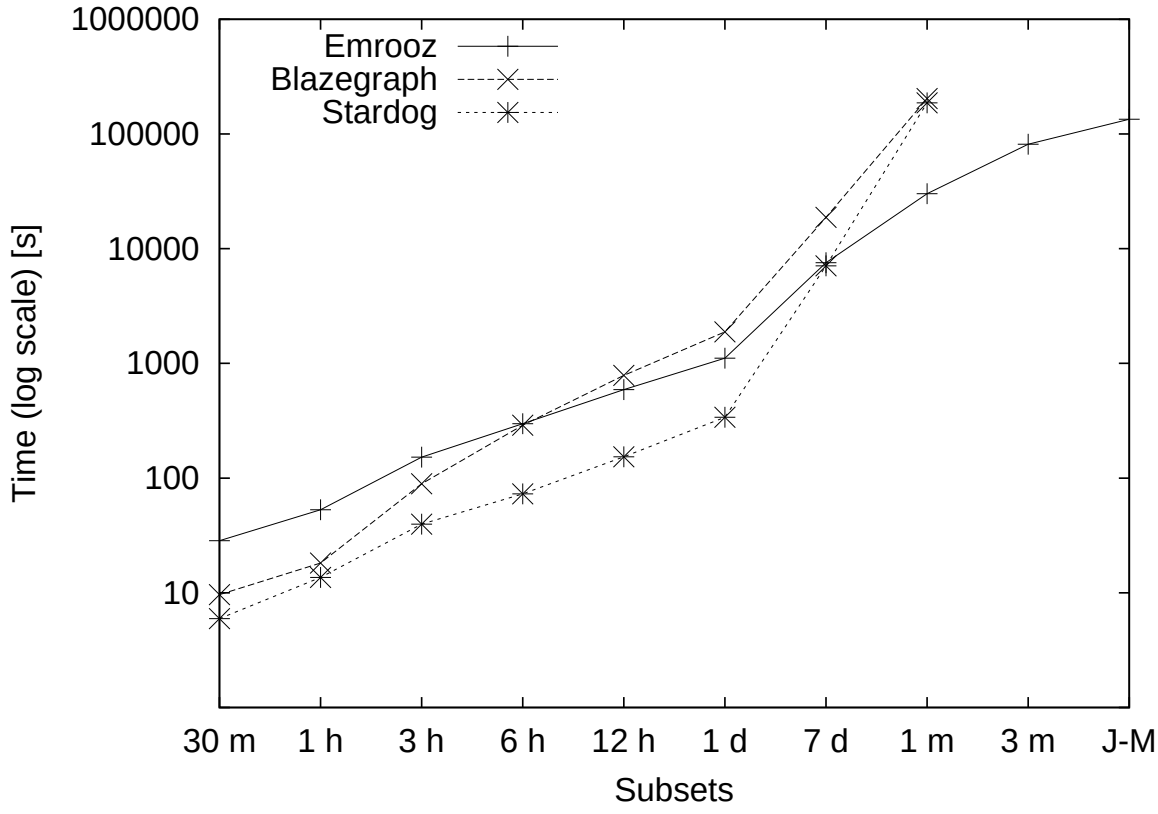
The query

```
select ?time ?value
where { [
  ssn:observedBy licor:LERS-75H-2035 ;
  ssn:observedProperty sweet-propFraction:MoleFraction ;
  ssn:featureOfInterest sweet-matrCompound:CO2 ;
  ssn:observationResultTime [ time:inXSDDateTime ?time ] ;
  ssn:observationResult [ ssn:hasValue [
    dul:hasRegionDataValue ?value
  ] ]
]
  filter (?time >= "2015-04-15T00:00:00.000+06:00"^^xsd:dateTime
    && ?time < "2015-04-15T00:10:00.000+06:00"^^xsd:dateTime)
}
order by asc(?time)
```

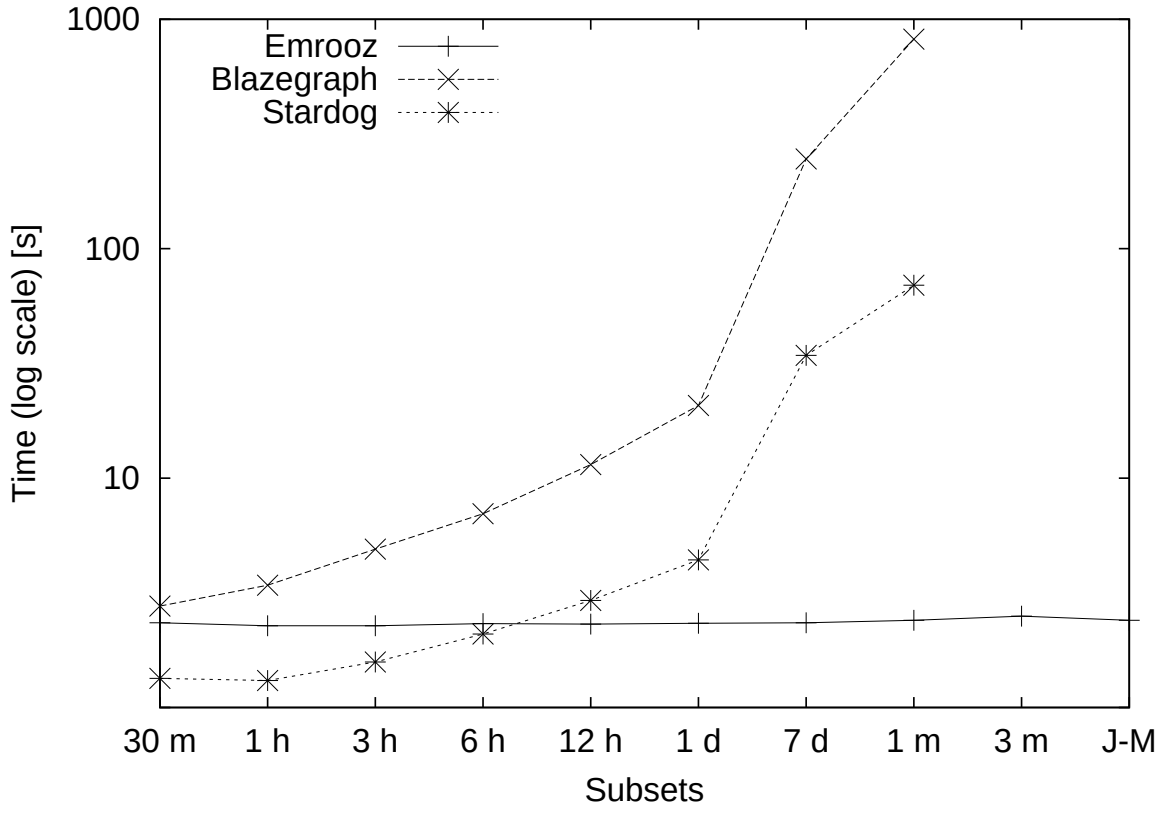
Results: Some figures

Subset	Observations	Triples	Distinct
30 m	54 000	810 000	648 007
1 h	108 000	1 620 000	1 296 007
3 h	324 000	4 860 000	3 888 007
6 h	647 997	9 719 955	7 775 971
12 h	1 295 997	19 439 955	15 551 971
1 d	2 591 994	38 879 910	31 103 935
7 d	18 140 271	272 104 065	217 683 259
1 M	72 526 464	1 087 896 960	870 317 575
3 M	194 188 107	2 912 821 605	*
J-M	328 715 445	4 930 731 675	*

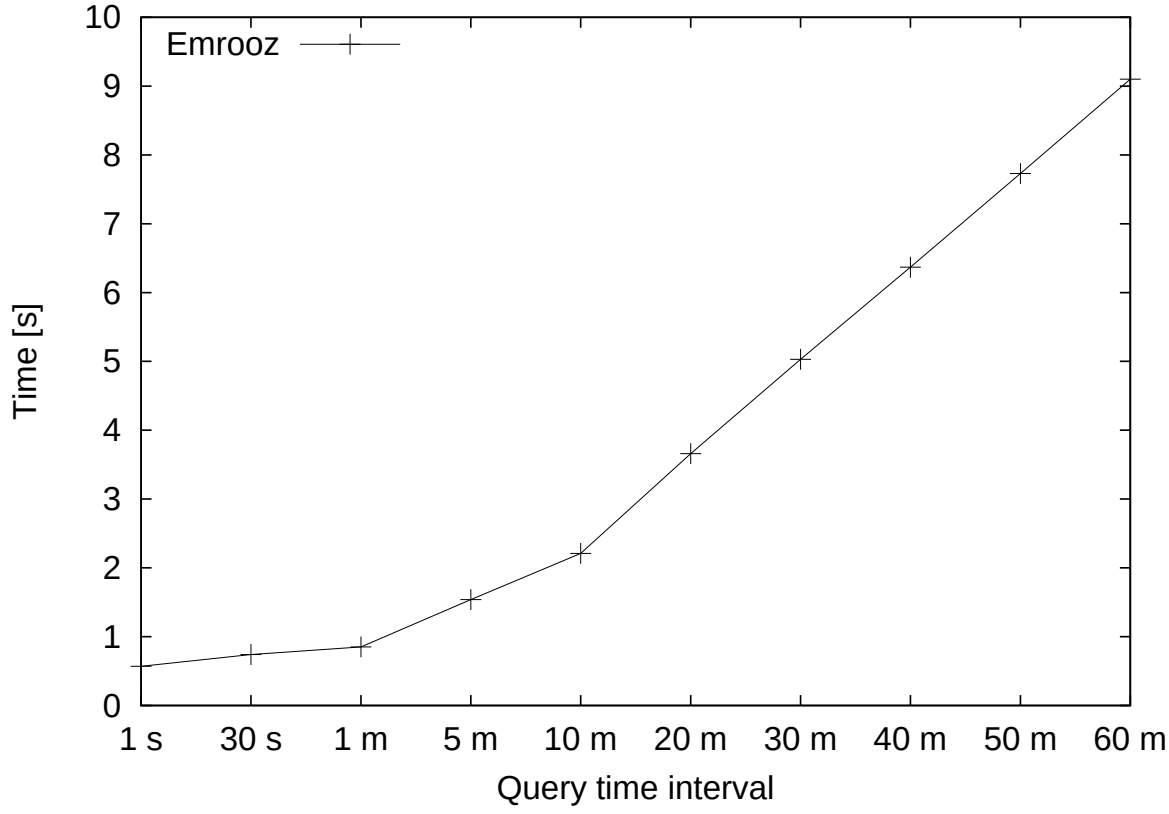
Results: Load performance



Results: Query performance



Results: Query size performance



REST

```
curl http://localhost:8080/sensors/list
curl http://localhost:8080/properties/list
curl http://localhost:8080/features/list

curl -H "Accept: application/json" \
  http://localhost:8080/sensors/list

curl -H "Accept: text/csv" -G \
  --data-urlencode sensor=http://example.org#thermometer \
  --data-urlencode property=http://example.org#temperature \
  --data-urlencode feature=http://example.org#air \
  --data-urlencode from=2015-04-21T01:00:00.000+03:00 \
  --data-urlencode to=2015-04-21T02:00:00.000+03:00 \
  http://localhost:8080/observations/sensor/list
```

R

```
host <- "http://localhost:8080"

df.sensors <- read.csv(text=getURL(paste0(host, "/sensors/list")),
  header=FALSE, col.names=c("sensor"))

df.sensors
                              sensor
1 http://licor.com#LERS-75H-CH4
2 http://licor.com#LERS-75H-CO2
```

R

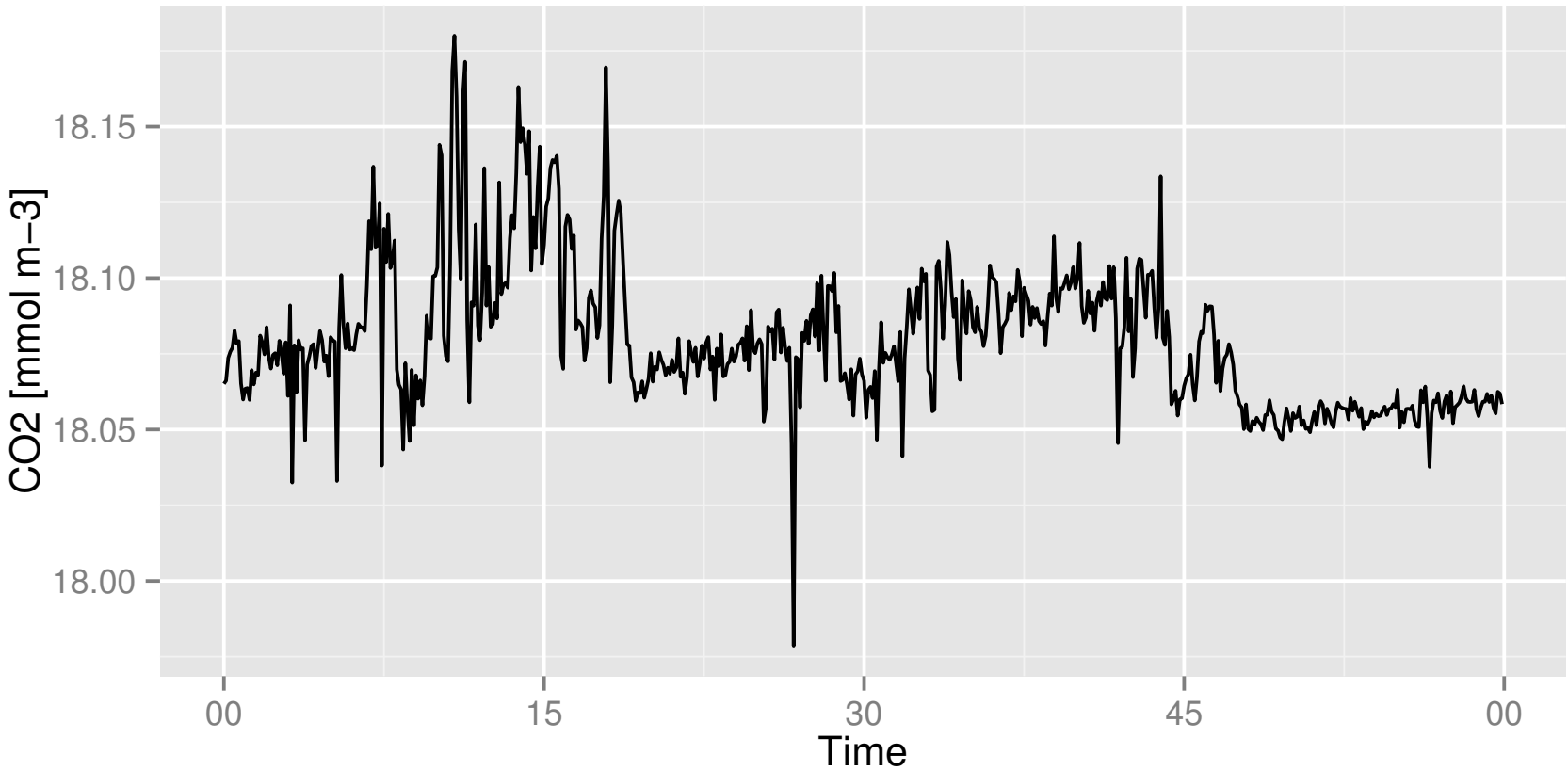
```
host <- "http://localhost:8080"
sensor <- "http://licor.com#LERS-75H-CO2"
property <- "http://sweet.jpl.nasa.gov/2.3/propMass.owl#Density"
feature <- "http://sweet.jpl.nasa.gov/2.3/matrCompound.owl#CarbonDioxide"
from <- "2015-01-07T00:00:00.000+06:00"
to <- "2015-01-07T00:01:00.000+06:00"

url <- paste0(host, "/observations/sensor/list?",
  "sensor=", curlEscape(sensor),
  "&property=", curlEscape(property),
  "&feature=", curlEscape(feature),
  "&from=", curlEscape(from),
  "&to=", curlEscape(to))

df.observations <- read.csv(text=getURL(url,
  httpheader=c(Accept="text/csv")), header=TRUE, sep=",")

ggplot(data=df.observations, aes(time, value))
  + geom_line() + xlab("Time") + ylab("CO2 [mmol m-3]")
```

R



Related and future work

- Other authors have pointed out the problem
- “Semantification of measurement data not promising”
- RDF databases on NoSQL systems (e.g. Cumulus RDF)
- Support for QB observations (done)
- REST API (preliminary)
- Integration with R/Matlab (preliminary)
- Performance comparison with other systems

Conclusion

- SSN and RDF nice for sensor (meta-) data
- Triple stores inadequate for observation data
- Alternative approaches required
- What are the advantages and disadvantages?
- Reasoning on all data by some sensor?
- Query for observation values exceeding threshold?