Applying Ontologies in the Dairy Farming Domain for Big Data Analysis

Jack P.C. Verhoosel and Jacco Spek

TNO Data Science, Soesterberg, The Netherlands
jack.verhoosel@tno.nl

Abstract. In the Dutch SmartDairyFarming project, main dairy industry organizations like FrieslandCampina, AgriFirm and CRV work together on better decision support for the dairy farmer on daily questions around feeding, insemination, calving and milk production processes. This paper is concerned with the inherent semantic interoperability problem in decision support information in a variety of big data sources containing static and dynamic sensor data of individual cows. Semantic alignment is achieved using ontologies and linked data mechanisms on a large amount of sensor data, such as grazing activity, feed intake, weight, temperature and milk production of individual cows at 7 dairy farms in The Netherlands. A Common Dairy Ontology (CDO) and a specific measurement ontology have been developed and used to transform 12GB of yearly sensor data into 350GB of RDF triples, made accessible via a SPARQL interface on the Apache Jena Fuseki triplestore. A few example applications have been developed to show how the CDO can be used for decision support and historic analysis. The performance of our linked data semantic solution is acceptable for analysis queries on large sets of data. Without optimization of queries the time for answering queries ranged from a few seconds to a couple of minutes.

1 Introduction

Dairy farmers are currently in an era of precision livestock farming in which information provisioning for decision support is becoming crucial to maintain a competitive advantage. Therefore, getting access to a variety of data sources on and off the farm that contain static and dynamic individual cow data is necessary in order to provide improved answers on daily questions around feeding, insemination, calving and milk production processes.

The process of selecting data sources was done with the wellbeing of the animals in mind. The sensors were applied to the animals, when necessary, by the farmers themselves. Also, these were non-invasive sensors like step-counters. Most sensors are external sensors attached to the machines that already interact with the animals (like milk-robots and feeding equipment). All data could be collected without causing additional stress or discomfort to the animals. The goal of the overall project in which this research was done was also to measure the improvement of the quality of life of the animals. The results can be used to better cater the individual needs of the cows, and be able to detect symptoms of illness of the animal, making a positive impact on their wellbeing.

In the Dutch SmartDairyFarming project, we work together with the main dairy industry organizations such as FrieslandCampina, AgriFirm and CRV, and use sensor

equipment to monitor cows at 7 dairy farms in The Netherlands. Thereby, a large amount of sensor data is generated on grazing activity, feed intake, weight, temperature and milk production of individual cows. A challenge in this sense is how to tackle the problem of semantic interoperability between the concepts present in dairy farming data sources. Semantic alignment of the different meanings of similar concepts in various data sources is therefore necessary for improved decision support and historical analysis.

We have focused on the use of ontologies and linked data mechanisms as a solution direction for this semantic interoperability problem. A Common Dairy Ontology (CDO) has been developed that serves as the main semantic interface to applications for dairy farming decision-making and analysis. Besides semantic alignment, the CDO also enables the reasoning on the dairy concepts and therefore on a large variety of different analysis questions. In addition, all sensor data has been transformed into triples according to the Resource Description Framework (RDF) standard[1], made available in a triplestore and accessible via a SPARQL engine[2]. With a few example applications we have measured the performance of this solution and assessed the feasibility and performance of our solution direction.

In the remainder of this paper we will describe subsequently our linked dairy sensor setup, the ontologies used for semantic alignment, the test applications and the performance of our setup.

The main contribution of our research is the practical insights in the challenges that arise when large amounts of sensor data are stored as triples. We have gained insights in performance behavior that arise when working with different technologies. We have also found some new challenges for future research, like how well do technologies that provide a mapping between traditional relational-databases (that are known to perform well with large amounts of data) and triples perform, and scale.

2 Related work

Sensors are a great tool for collecting near real-time data, vast networks of sensors are collecting more data than can be processed[1]. The sensor data can be enriched with semantic metadata to increase the possibility for interoperability between sensor networks and can also provide contextual information to the sensor data. Semantic web technologies can help to achieve this and can aid in the discovery and integration of new data sources. There have been efforts to enrich sensor data with semantic metadata in several domains.

In the agriculture domain, it is critical to monitor various environmental parameters (for example temperature, moisture, pH and electric conductivity) to sustain the best environment for the growth of plants. With the help of semantic technology like the semantic web in combination with semantic data integration, alerts can be set to signal the farmer of any issues regarding the attributes[2].

[1] https://www.w3.org/RDF/
[2] http://www.w3.org/TR/sparql11-query/

In the industrial building context, there is great value in a high quality alerting system, since the cost of fixing damages is significantly higher than implementing an early warning system that signals the possibility of damage happening to a building. Such a system, using sematic sensor technologies is proposed in [3] and allows for notifications to be sent in case of water leaks. But also when a dangerous combination of factors occur, like the buildup of pressure and temperature, which could indicate a fire.

3 Linked dairy sensor data

At each of the 7 dairy farms involved in our SmartDairyFarming project[3] sensor equipment is installed to monitor an average of 400 cows per dairy farm. These cows are continuously monitored since 2014, which has generated a yearly 12GB of sensor data. This sensor data is made available via a software component called the InfoBroker to be used by various different applications for the dairy farmer. Our approach is to transform this sensor data into linked data in order to make it semantically rich and easily accessible.

To do this in a methodological way, we have developed a Linked Data Roadmap. This roadmap is developed to standardize the process of converting data of different types into linked data. In this roadmap we have defined nine steps containing tools- and best practices to generate high quality linked data. The Linked Data Roadmap is visualized in **Fig. 1**. For the sake of simplicity we will not describe the details of the roadmap here and further details can be found at the website of PDLN[4].

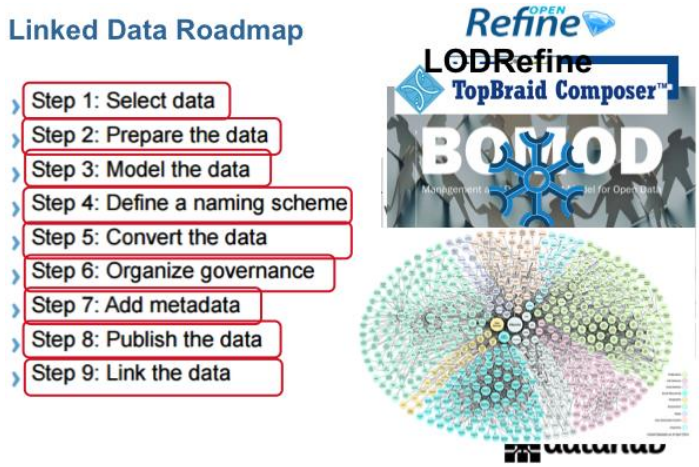

Fig. 1. Linked Data Roadmap with steps and tools to convert data into linked data.

We applied the process defined in this roadmap to make the data that is currently available through the InfoBroker available as linked data. Thereby, it becomes possi-

[3] http://www.smartdairyfarming.nl
[4] http://www.pilod.nl/wiki/BoekTNO/stappenplan

ble to query this data using semantic queries and make use of the advantages that this technology offers.

For the second step of the roadmap on preparing the data, we needed to extract all the data from the InfoBroker. Because the InfoBroker offers its data as a REST-API, we designed a python script that extracts data from the InfoBroker by calling a sequence of API calls, and transforms these JSON responses into CSV files. We then used GoogleRefine[5] to determine the quality of the data to make a clean and consistent dataset.

Based on this clean dataset we applied step three of the roadmap on modeling the data. Modeling data in terms of linked data means defining an ontology or knowledge model for the data. This can be seen as the database schema, it defines the meaning of things in terms of relationships. For this case, we defined a specific measurement ontology based on three basic classes: *Cow, Sensor* and *Measurement*[6]. This ontology is depicted in **Fig. 2**.

With this ontology it was possible to express all the data offered by the InfoBroker in terms of triples of the form <*subject, predicate, object*>. We then defined a sensible naming convention for the URI's, so that all URI's would be unique and meaningful.

Finally, we've developed a python script that automates the process of converting the CSV files to RDF files. This script uses the API of GoogleRefine to automatically convert the CSV to RDF triples. This script uses earlier defined operations on the CSV data (including the definition of the RDF structure). This script then exports the GoogleRefine projects as RDF files, which it then uploads to our triplestore, so expose the data and make it queryable using the included SPARQL endpoint. The yearly 12GB sensor data was transformed into 310GB of triple data.

[5] http://openrefine.org
[6] http://minion02.sensorlab.tno.nl/ontologies/SDF.ttl

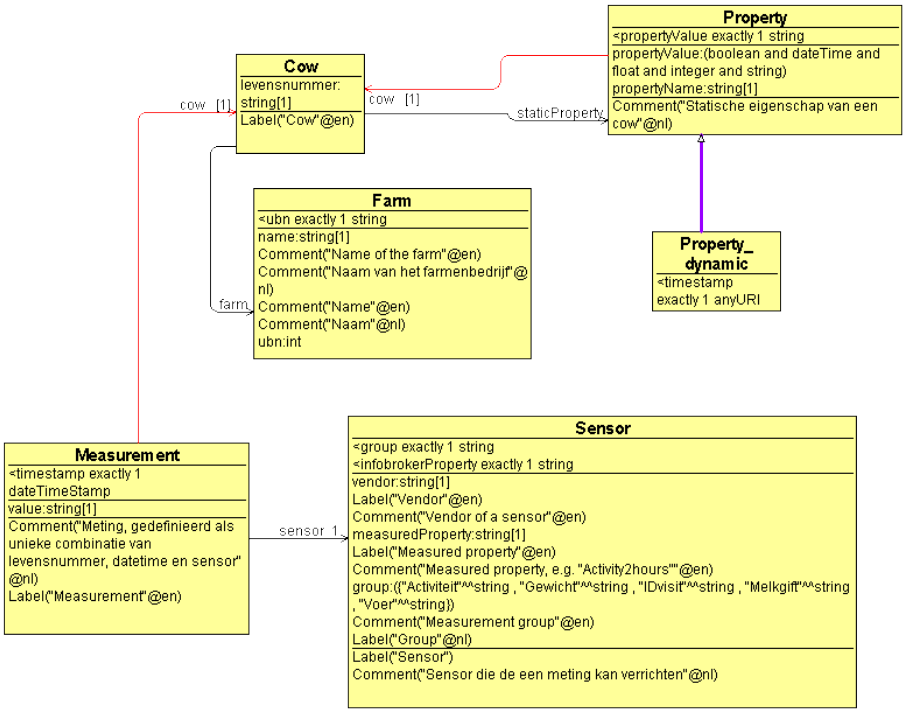

Fig. 2. A measurement ontology for modeling measurements of sensor parameters for cows.

We've chosen to use Apache Fuseki[7] as our triplestore and SPARQL endpoint. It uses Apache TDB (Triple DataBase) which is a graph database for storing of triples. Apache Fuseki in combination with the Apache TDB triplestore offers advanced SPARQL support, including federated queries, as well as a very high performance compared to for instance Apache Marmotta[8]. The linked data is queryable through the Apache Fuseki SPARQL endpoint. This data is kept up-to-date with a fully automated process that daily extracts data from the InfoBroker and stores it as triples in the Apache Fuseki store.

4 Semantic alignment

One of our goals is to enable the answering of analysis questions on the combination of large sources of measured sensor data. One of these questions is for instance *"What is the average weight per day over the last lactation period of a cow of a farmer?"* In order to achieve this, we have developed a Common Dairy Ontology (CDO)[9] that is meant to contain the main, common dairy farming concepts. The CDO

[7] https://jena.apache.org/index.html
[8] http://marmotta.apache.org
[9] http://minion02.sensorlab.tno.nl/ontologies/cow-model.ttl

ontology includes concepts like *Farm, Farmer, Cow, Weight, Milkyield, Activity, Feed, Parcel, Equipment* and so on. In addition, the CDO ontology covers the most important relationships between these concepts. The CDO then functions as the "semantic interface" to the users of the information that is captured by the sensor data system. The users can be farmers, advisors of farmers or other stakeholders around the farm and they want to express their information needs in terms of these common concepts in the CDO. See **Fig. 3** with an excerpt of the concepts in the CDO.

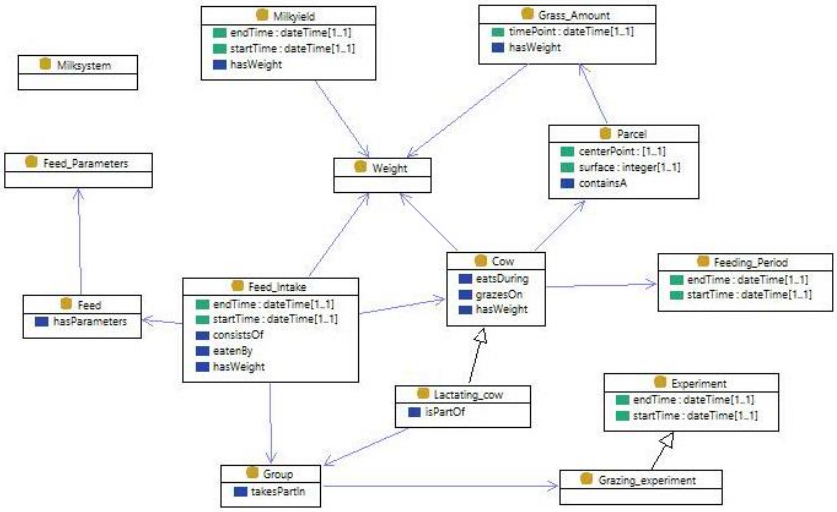

Fig. 3. Common Dairy Ontology excerpt with main concepts in dairy farming.

The CDO can be used as the knowledge model for accessing the sensor data that has been modeled as triples using the specific measurement ontology. In order to do this, we have made a mapping from the CDO to the specific measurement ontology and the specific parameters that are measured by the sensor equipment. For example, the common concept *Weight* is mapped onto various different parameters for weight measured by different weighing equipment at the farm, such as *Lely.BodyWeight* and *GallagherDairyScale.dsweight*. In addition, the concept Activity is mapped onto various specific parameters for activity, such as *Lely.Activity2hours* and *DeLaval.HoogActNiveau*. We used the rdfs:label mechanism to make the mapping between the classes and properties of the CDO and the measurement ontology.

See **Fig. 4** with an example of the mapping between the CDO and the measurement ontology. This mapping is being used when big data analysis question are asked to the CDO for which measured sensor data is needed.

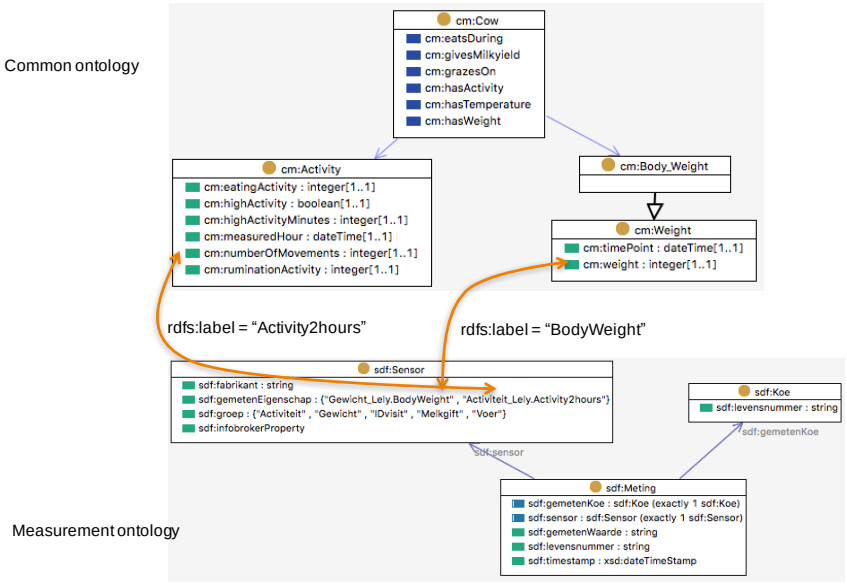

Fig. 4. Mapping of classes and properties between the CDO and the measurement ontology.

5 Big data applications

In order to use the large set of historical sensor data and to assess the performance of the triple solution, big data applications have been developed that focus on the analysis of possible patterns in the data of individual cows over the period of one year.

One application looks at the relation between bodyweight and milk yield of individual cows during the lactation period in 2014. For each individual cow the development of bodyweight and milk yield during the lactation period can be drawn in a graph. In addition, the increase/decrease of the bodyweight on a weekly basis during that same period can be drawn as well. Finally, an overall view of the average weight over all the cows of the same parity can be depicted. Using these views, the farmer can derive possible relationships between the bodyweight and the milk yield. Another application tries to find similar relations between the different types of feed and milk yield of individual cows during the lactation period in 2014. For each individual cow the development of total feed intake and milk yield during the lactation period can be drawn in a graph. In addition, the division of the total feed intake over various feed types during that same period can be drawn as well. Finally, an overall view of the total feed intake over all the cows of the farmer can be depicted. Using these views, the farmer can derive possible relationships between the intake of different types of feed and the milk yield. See **Fig. 5** for a view on the front end of the applications.

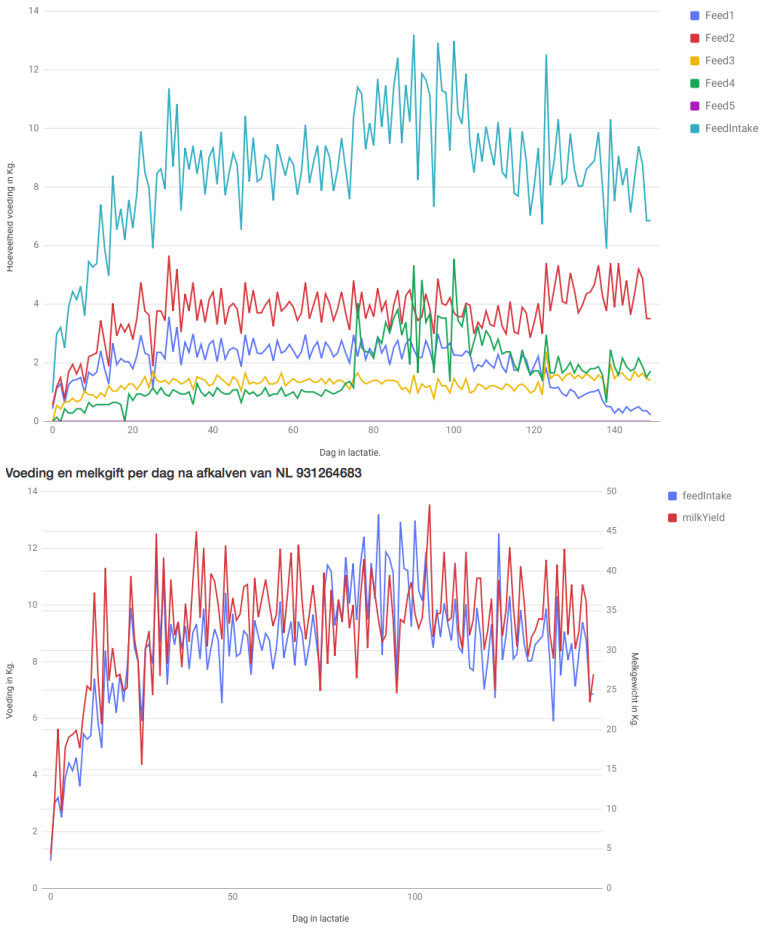

Fig. 5. Big data analysis applications on historical weight, feed and milk yield data.

Each of the graphs that can be shown by the applications is build up based on the result of a SPARQL query on the large set of triples. The SPARQL queries use FILTER statements to select those triples that are related to measurements for the specific cow number, the correct time period and the measured parameters. See **Fig. 6** for the SPARQL query code for two of the queries that we used to select feed and weight data from the Fuseki triplestore.

6 Performance

We've started our experiments with the Apache Marmotta triplestore including a SPARQL endpoint. However, it turned out that our large volume of data had a significant impact on the performance of the Marmotta triplestore.

```
# Feed per day per type over all cows of a farmer

PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX minion: <http://minion02.sensorlab.tno.nl:8080/fuseki/SDF/data/>
PREFIX sdf: <http://minion02.sensorlab.tno.nl/ontologies/SDF.ttl#>
SELECT ?type ?dag (SUM(xsd:float(?w)) AS ?waarde)
FROM minion:%s #Machine
WHERE {
  ?sensor sdf:gemetenEigenschap ?type .
  FILTER regex(?type, "Feed") .
  ?meting sdf:sensor ?sensor .
  ?meting sdf:timestamp ?ts .
  FILTER( xsd:dateTime(?ts) >= xsd:dateTime("2014-01-01T00:00:00.0") &&
xsd:dateTime(?ts) < xsd:dateTime("2015-01-01T00:00:00.0"))
  ?meting sdf:gemetenWaarde ?w .

  #calculate JND of ?ts
  BIND(floor((14-month(?ts))/12) as ?a_ts).
  BIND(year(?ts) + 4800 - ?a_ts as ?y_ts).
  BIND(month(?ts) + (12*?a_ts) - 3 as ?m_ts).
  BIND(day(?ts) + floor((153*?m_ts + 2) / 5) + (365*?y_ts) + floor(?
y_ts/4) - floor(?y_ts/100) + floor(?y_ts/400) - 32045 - 2456658 as ?
dag).

}
GROUP BY ?type ?dag
ORDER BY ?type ?dag
```

```
# Average weight of a cow in a lactation period

PREFIX fuseki: <http://minion02.sensorlab.tno.nl:8080/fuseki/SDF/data/>
PREFIX sdf: <http://minion02.sensorlab.tno.nl/ontologies/SDF.ttl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
SELECT DISTINCT ?parity ?date (ROUND(AVG(xsd:integer(?waarde))) AS ?gewicht)
FROM fuseki:Antonides_Lely
FROM fuseki:Antonides
WHERE {
       ?koe sdf:levensnummer "%s". #Cow
       ?staticProp sdf:koe ?koe.
       ?staticProp sdf:eigenschapNaam "calvingDate".
       ?staticProp sdf:eigenschapWaarde ?calvingDateStr.
       BIND(xsd:dateTime(?calvingDateStr) as ?calvingDate)
       ?staticProp2 sdf:koe ?koe.
       ?staticProp2 sdf:eigenschapNaam "dryDate".
       ?staticProp2 sdf:eigenschapWaarde ?dryDateStr.
       BIND(xsd:dateTime(?dryDateStr) as ?dryDate).
       ?staticProp3 sdf:koe ?koe.
       ?staticProp3 sdf:eigenschapNaam "parity".
       ?staticProp3 sdf:eigenschapWaarde ?parityStr.
       BIND(xsd:int(?parityStr) as ?parity)
       ?sensor sdf:gemetenEigenschap "BodyWeight" .
       ?meting sdf:sensor ?sensor .
       ?meting sdf:gemetenKoe ?koe .
       ?meting sdf:timestamp ?ts.
       FILTER( ?ts >= xsd:dateTime(?calvingDate)) .
       ?meting sdf:gemetenWaarde ?waarde .

       BIND ( CONCAT( STR(YEAR(?ts)),"-",STR(MONTH(?ts)),"-" , STR(DAY(?
ts)) ) AS ?date).
}
GROUP BY ?parity ?date
ORDER BY ?ts
LIMIT %d #Number_of_days
```

Fig. 6. Two SPARQL queries used to select feed and weight data from the Fuseki triplestore.

The bad performance of the Marmotta triplestore was likely due to the fact that Marmotta stores the data in a relational database, and translates this to triples when it's queried. We tried to optimize the Marmotta and the underlying PostgreSQL database configuration. However, performance remained unacceptable for our purposes. Another drawback of the Marmotta server is the limited support of the full SPARQL1.1 standard and federated queries. This was a functionality we wanted to have to be able to query the data combined with other datasets offered through other SPARQL endpoint. Apache Fuseki performed a lot better in that sense. This is most likely because Fuseki implements a native triple graph database, instead of a relational database that converts data into triples.

Query	***Input***	***Graph size***	***Search par***	***Response***
Select an overview with the number of cows of a farmer	Farmer-S	111,604,625	1	0.04s
	Farmer-B	167,894,559	1	0.03s
	Farmer-A	79,739,365	1	0.37s
Select the list of cows with number and parity	Farmer-S	28,704	3	0.934s
	Farmer-B	9,400	3	15.110s
	Farmer-A	45,816	3	27.006s
Select feed per type per day over all cows of a farmer	Farmer-S	66,551,765	3	913.003s
	Farmer-B	38,034,692	3	350.917s
	Farmer-A	45,637,592	3	380.470s
Select average weight over all cows per day per parity	Farmer-A	45,637,592	3	348.704s
Select static info for a cow	NL 715820911	45,816	2	0.094s
Select weight per day in lactation period	NL 715820911	45,683,408	5	5.129s
Select weight and milkyield per day in lactation period	NL 715820911	45,683,408	7	13.714s
Select milkyield per day in lactation period	NL 715820911	45,683,408	3	4.142s

Fig. 7. Overview of graph size, selection parameters and response times for the main queries.

We did a number of measurements on the response times for the SPARQL queries executed in our application. In **Fig. 7** we present an overview of the queries, the input parameters (farmer or cow), the graph size in number of triples in which the query is executed, the number of search parameters in the query and the response time in seconds.

An interesting observation is that there is hardly any pattern to be recognized between the graph size, the number of search parameters and the response times. Apparently, the SPARQL search engine has specific ways of indexing to assist the search process through the set of triples. The performance of our linked data semantic solution is acceptable for analysis queries on large sets of data. Without optimization of queries the time for answering queries ranged from a few seconds to a couple of minutes. This is acceptable for analysis purposes, but for analyzing real-time data, this is probably not acceptable. The more complex queries take several minutes to generate a response (this could even take up to 30 minutes for a complex query when the server-load is high). A farmer might not be willing to wait that long for a response.

7 Conclusion and future work

The overall conclusion of our work presented in this paper is that the use of linked data mechanisms on large sets of sensor data is feasible. Even if big data in the order of hundreds of gigabytes is put into RDF triple format, it remains accessible with reasonable response times for analysis purposes. For near real-time purposes the question remains whether this solution approach is still feasible, but this is a topic for further work. Another future work topic is to further extend the Common Dairy Ontology with more dairy concepts and make it available as the semantic interface in conjunction with the InfoBroker or a local relational database. In that setup the sensor data is not being put into RDF triples and the CDO is used as mapping tool towards the specific sensor data sources.

8 References

[1] L. Lefort, K. Janowicz, P. Barnaghi, O. Corcho, J. Graybeal, A. Herzog, A. Nikolov, K. Page, R. G. Castro, and M. Compton, “Semantic Sensor Network XG Final Report.” 2011.

[2] A. Sheth, C. Henson, and S. S. Sahoo, “Semantic sensor web,” *IEEE Internet Comput.*, vol. 12, no. 4, pp. 78–83, 2008.

[3] G. P. Zarri, L. Sabri, A. Chibani, and Y. Amirat, “Semantic-Based Industrial Engineering: Problems and Solutions,” *Complex, Intelligent and Software Intensive Systems (CISIS), 2010 International Conference on*. pp. 1022–1027, 2010.