Good afternoon, everybody.

I am Li Qiong, and it is a great pleasure for me to give a lecture for all of you.

二

My title is “an adaptive framework for RDF stream processing.”. I will present my paper from the following aspects. ---- the introduction, RDF stream and C-SPARQL, the reduction from C-SPARQL to SPARQL, our framework and our experiments and evaluations.

三

It is a stream world! RDF stream, as a new type of dataset, can model real-time and continuous information in a wide range of applications, e.g. environmental monitoring and smart cities.

Let us see what happens in an Internet minute. We can see that 47,000 Apps are downloaded, 6 million facebook views, more than 2 million search queries in Google, 639,800 GB of global IP data transferred, and so on.

四

Certainly, there are a few works by extending SPARQL to support queries over RDF streams, such as C-SPARQL, EP-SPARQL, CQELS, etc. But it is difficult to implement these engines.

However, there are many SPARQL query engines processing RDF graphs, such as the centralized engines, Sesame, Jena, RDF-3X, gStore, and distributed engines, gStroeD and TriAD and so on. But these works are only for processing RDF graphs which change infrequently.

五

So, it is an interesting problem about how to design a framework which can apply the current SPARQL query engines to process RDF streams. And the framework will make it easier to process RDF streams.

六

Firstly, we briefly introduce the RDF stream and C-SPARQL.

An RDF triple is formed by a triple (s, p, o). And an RDF graph is a finite set of RDF triples.

An RDF stream is defined as an ordered sequence of pairs which are quadruples, and each pair is made of an RDF triple (s, p, o), and a timestamp t.

This is the definition of RDF stream.

$$S(t) = \{\langle s_i, p_i, o_i \rangle, \tau_i) \mid \tau_i \leq \tau_{i+1}\}.$$

And now, we give an example of RDF stream.

It is a stream from the time of 1483850233586 to 1483850237596. The stream shows the temperature values from the environmental monitoring sensors.

七

And then we introduce the C-SPARQL.

C-SPARQL is the continuous SPARQL, which extends SPARQL by adding new operators, namely, registration and windows, to support processing RDF streams.

The following is the syntax of some operators of C-SPARQL.

These are the operators of registration of query and stream.

C-SPARQL queries should be continuously registered to provide continuous querying services.

Query Registration gives the query name which will be processed.

The same as Stream Registration. The difference is that query registration will return the result the query needs, and the stream registration will return a new RDF stream.

八

This is the window operator, a very important operator. The window is defined in the FROM clause. We can see that windows depend upon two parameters, namely, the window size (RANGE) and the window step (STEP).

Now, we give an example of C-SPARQL.

The query shows the different temperature in different observation at any time. The query name is SensorQuery, and the window size is 5 seconds, the window step is 4 seconds.

九

Now, let us think about the question. If we want to exploit the SPARQL query engines to process RDF streams, we must reduce the C-SPARQL query to the SPARQL query.

So, the next I will give the formal definition of both C-SPARQL and window, and then prove them to ensure the soundness and completeness of our reduction.

十

A C-SPARQL query Q can be taken as a 5-tuple of the form:

Where, Req is the registration clause, the uppercase S is the RDF stream which will be registered. W is the window size, the lowercase s is the updating time of windows, and ρ(Q) is the SPARQL query.

十一

Now, we give an example of C-SPARQL that is mentioned before. We can obtain that Req is the first line, S is the stream SensorStreams which is defined in the from clause. And w equals 5 seconds, s equals 4 seconds, ρ(Q) is the standard SPARQL query as follows. I will not explain it here.

十二

Now, we give a brief analysis about the difference from C-SPARQL to SPARQL.

First, the registration of C-SPARQL query ensures the continuous recall of C-SPARQL query periodically. However, SPARQL query does not support such a continuous mechanism for recalling query.

Second, C-SPARQL can support RDF streams but SPARQL cannot due to the timestamp of tuples of RDF streams.

Third, C-SPARQL query consists of RDF streams which are to be processed.

十三

Based on discussions above, we can see that window is an important notion in transforming C-SPARQL to SPARQL.

Now, this is the definition of window. The uppercase S is the RDF stream, w is the window size, the lowercase s is the updating time of windows, and k is a integer which denotes the k-th window.

十四

Now, we give an example of windows, the initial window and the first window. And the windows are produced from the RDF stream Sensor.

十五

For any RDF stream S, the results from the continuous query Q over streams are equal the results from the query Q over the windows that are produced from the stream.

And the results from the query Q over the windows are equal the results from the SPARQL query ρ(Q) over the RDF graphs which are transformed from the windows.

So, we can get that the results from the continuous query over streams are equal the results from the SPARQL query over the RDF graphs.

十六

Now, I will introduce our framework.

Our framework contains four main modules, query parser, trigger data transformer, and SPARQL API.

The query parser will transform the query from C-SPARQL to SPARQL.

The trigger calls the SPARQL query and produce window selector periodically.

The data transformer will transform the data from RDF streams to RDF graphs.

And the SPARQL API will process the RDF graphs obtained before. And so far, our framework has support three centralized engines, Jena, RDF-3X and gStore, and two distributed engines, gStoreD and TriAD.

十七

Now, I will briefly introduce our experiments and evaluations. And if you want to know the details, you can read our paper.

We do our experiments by utilizing YABench RSP benchmark. And we compare the different SPARQL query engines in processing RDF streams under different input loads.

We can find that the capacity of both loading data and processing data of SPARQL query engines influence the capacity of them to process RDF streams.

十八

Above all, our major contributions are summarized as follows:

Firstly, we formalize a C-SPARQL query to ensure the soundness and completeness of our reduction from C-SPARQL queries to SPARQL.

Secondly, we develop an adaptive framework for processing RDF stream, and support distributed SPARQL query engines to process large-scale RDF streams.

And last, we implement the framework and we can evaluate the different SPARQL query engines in processing RDF streams in a unified platform. We also exploit the YABench to evaluate the results.

That is all. Thank you for your attention.