Spatial Searches with SPARQL using Apache Jena

Andy Seaborne, Ying Jiang
Apache Software Foundation

Apache Jena is a mature, open source, linked data framework. It can store and query data scaling up to many hundred's of millions of triples. It provides SPARQL 1.1 as the way to query the data, including over the web using HTTP.

It also provides additional indexing of data beyond just RDF storage. The most common use of this feature previously has been text search of document content. The text index service provides a way of finding which URIs answer a text search. The content indexed can be part of the RDF data but this is not a requirement. Documents can be indexed by their text content, and only some of the metadata of the original documents stored in Jena in RDF.Text indexing is provided by Apache Lucene or, for larger scale deployments, Apache Solr. The index can be shared with other applications.

Can this approach of additional specialized access be applied to geospatial data? That is, can a specialised index provide a way to find points of interest based on geospatial relationships.

At the same time, we wish to provide capabilities to general web developers, not just geospatial experts. Indeed, the sheer complexity of geospatial data is potentially off-putting because the impression is that one needs to be an expert in order to do any operations, however simple. Yet the general web developer is overrun by different technologies competing for their time.

A Simple Use Case

Jena Spatial is an additional indexing system that is targeting just point data. The canonical use case is how to find linked data entities that are close to some point. Having found some points of interest, additional data can be added, taken from the RDF linked data database.

This is the basis of the "putting markers of a map" problem. It should not require the application web developer to be an expert in geospatial systems.

Need for Something

Previous experience of processing bounding box queries without specific index support showed that for moderate sized data sets, tolerable performance could be achieved just with plain SPARQL. In outline:

```
SELECT * {
    ?x :easting  ?east .
    ?x :northing ?north .
    FILTER( ?east > ?? && ?east < ?? && ?north > ?? && ?north < ?? )
}
```

but without additional indexing on values, this query can result in the same amount of work regardless of box size. It does not account for the fact that the earth is not flat. Circle calculations would require more work in query processing and require the web application developer to write out the equations.

Simple Implementation Required

GeoSPARQL provides a comprehensive approach to querying geospatial data. The implementation of a complete GeoSPARQL solution is non-trivial both in the scale of the work and the impact on the query engine design, and hence the whole system, because it is based on a set of rules for query transformations. This needs to be put into the heart of the query processing workflow, making the query engine more complicated to maintain for general purpose use.

We decided to provide an external index in the style of free text search and combine it with the stable, performant, but geo-unaware, query engine.

Spatial Index

The spatial index used is Apache Lucene Spatial. The practical advantage is that Jena already uses Lucene for text indexing so does not require another system to be incorporated and also it can easily be change to use Apache Solr (based on Lucene) to give additional scale and performance. The approach is general and, unlike the text search feature, does not expose the query language of the index. Other indexes could be used.

While this is not a full geospatial solution (no "places near a road"), we hope it provides some easy capability for some usages.

Indexes : finding things

An index is something that matches a request, composed of a number of parameters, to a list of results (zero or more).

In SPARQL, the way to do this is in the basic graph pattern matching. It is the only place in the language where matching occurs.

SPARQL has an extension mechanism - custom functions - but a SPARQL function returns one result so it can't return all the matches (zero or more) to some index lookup. The RDF data model does not have lists as values (RDF lists are collections of triples).

At the same time, experience with other syntactic extensions of the SPARQL language have not proven to be popular. Users prefer to remain within the standard syntax even when the capabilities are extensions.

Spatial search

This example SPARQL query finds all points with 10 (UK statute) miles of Bristol, UK. The access to the spatial index is triggered by the `spatial:nearby` property. Additional information is included in the query results with the use of `rdfs:label`.

```
PREFIX : <http://example/>
PREFIX spatial: <http://jena.apache.org/spatial#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT * {
    ?s spatial:nearby (51.30 -2.71 10 'miles') ;
      rdfs:label ?label
}
```

This shows the common issue of mixing units. Bristol, UK is latitude 51.30, longitude -2.71 as might be used with the W3C WGS84-based vocabulary. Distance is expressed in the local units of "miles". The spatial query extension translates between different unit systems.

`spatial:withinCircle` is a synonym for `nearby`. Other matches include `spatial:withinBox` and being about ask for places "north of" and "east of" etc.

Availability

Available as part of Apache Jena.

This work was done as part of the Google Summer of Code 2013. We are grateful to Google for providing the funding for Ying Jiang.