tSPARQL: Using Quadstores for Temporal Querying of RDF

Overview

There are many existing RDF vocabularies that allow temporal data to be encoded in RDF. However, directly querying this temporal data is difficult in SPARQL for two reasons:

• Attributes that semantically belong to an object are encoded as belonging to a temporal "condition".
• Multiple encodings of temporal data means multiple query paths are necessary to retrieve similar data that have been encoded differently.

This project proposes an extension to the SPARQL query language that allows consistent querying of temporal data.

Solution

By utilizing the quadstore ability of most RDF “triplestores”, temporal information may be stored with each triple for querying. On import, triples are converted to temporal quads:

Two variants of a new SPARQL graph pattern ("TIME") are introduced for querying temporal data:

```
TIME ?interval { ... }
TIME [ a time:Interval ; ... ] { ... }
```

Similar to SPARQL’s GRAPH pattern, TIME allows triple matching to be restricted to time intervals matching given criteria.

The query "Find all people who lived in Tokyo during 2007" now becomes:

```
SELECT ?name WHERE {
  TIME [ time:inside "2007"^^xsd:dateTime ] { 
    [ a foaf:Person ; foaf:name ?name ;
      whois:place "Tokyo, Japan" . ]
  }
}
```

Example Data

Non-temporal data:

```
_:albert a foaf:Person;
  foaf:name "Albert Einstein".
```

Interval data (using the bio vocab1):

```
_:pauline a foaf:Person;
  bio:condition [ a bio:Condition;
    time:begins "1858";
    foaf:name "Pauline Koch" . ];
  bio:condition [ a bio:Condition;
    time:ends "1876";
    foaf:name "Pauline Einstein" . ];
```

Interval data (using the whois vocab2):

```
_:kanzaki a foaf:Person;
  whois:stage [ a whois:Stage;
    whois:place "Tokyo, Japan";
    whois:since "1982" . ];
  whois:stage [ a whois:Stage;
    whois:place "Mie, Japan";
    whois:born "1960";
    whois:until "1978" . ];
```

Temporal Import turns RDF Triples into Temporal Quads.

Temporal Inference Engine generates temporal relationship such as: before, inside, overlaps, contains, during.

tSPARQL Engine executes temporal queries, using temporal quads to answer queries about temporal information. Temporal data is ignored for simple (non-temporal) triple pattern matching.

References:
