6-16-2009

Extending SPARQL to Support Spatially and Temporally Related Information

Prateek Jain  
*Wright State University - Main Campus*

Amit P. Sheth  
*Wright State University - Main Campus*, amit.sheth@wright.edu

Peter Z. Yeh

Kunal Verma  
*Wright State University - Main Campus*

Follow this and additional works at: [https://corescholar.libraries.wright.edu/knoesis](https://corescholar.libraries.wright.edu/knoesis)

Part of the [Bioinformatics Commons](https://corescholar.libraries.wright.edu/knoesis), [Communication Technology and New Media Commons](https://corescholar.libraries.wright.edu/knoesis), [Databases and Information Systems Commons](https://corescholar.libraries.wright.edu/knoesis), [OS and Networks Commons](https://corescholar.libraries.wright.edu/knoesis), and the [Science and Technology Studies Commons](https://corescholar.libraries.wright.edu/knoesis)

Repository Citation

[https://corescholar.libraries.wright.edu/knoesis/672](https://corescholar.libraries.wright.edu/knoesis/672)
Extending SPARQL to Support Spatially and Temporally Related Information

Prateek Jain, Amit Sheth
Kno.e.sis, Wright State University, Dayton, OH

Peter Z. Yeh, Kunal Verma
Accenture Technology Labs, San Jose, CA
Increased availability of spatial information
But accessing this information can be difficult
User expected to ask for this information in the “right” way
Proposed approach

Automatically align conceptual mismatches between a user’s query and spatial information of interest through a set of semantic operators. Our approach will reduce the user’s burden of having to know how information of interest is structured, and hence improve accuracy and relevance of the results.
Outline

• Introduction
• Existing Mechanisms for querying RDF Data
  – Existing approaches
  – How well do they work?
• Proposed Approach
• Future Work
Why is it important?

- Spatial data becoming more significant day by day.

- Crucial for multitude of applications:
  - GPS
  - Military
  - Location Aware Services
  - weather data...

- Spatial Data availability on Web continuously increasing.
  - Sensor streams, satellite imagery
  - Naïve users contribute and correct spatial data too which can lead to discrepancies in data representation.
    - E.g. Geonames, Wikimapia
What’s the problem

- Existing approaches only analyze spatial information and queries at the lexical and syntactic level.

- Mismatches are common between how a query is expressed and how information of interest is represented.
  - Question: “Find schools in NJ”.
  - Answer: Sorry, no answers found!
  - Reason: Only counties are in states.

- Natural language introduces much ambiguity for semantic relationships between entities in a query.
  - Find Schools *in* Greene County.
What needs to be done?

- We need to reduce users’ burden of having to know how information of interest is represented and structured in order to enable access to this information by a broad population.

- We need to resolve mismatches between a query and information of interest due to differences in granularity in order to improve recall of relevant information.

- We need to resolve ambiguous relationships between entities due to natural language in order to reduce the amount of wrong information retrieved.
Existing mechanism for querying RDF
Known approaches

- SPARQL
- Path Expressions
Common query for testing all approaches!

“Find schools located in the state of Ohio”
In a perfect scenario
In a not so perfect scenario

Ohio \(\rightarrow\) Contains feature \(\rightarrow\) County \(\rightarrow\) Contains feature \(\rightarrow\) School
And finally..
Let's test the approaches!
SPARQL

- SPARQL Protocol and RDF Query Language.

- User to express queries over data stores where data stored as RDF or viewed as RDF.

- W3C Recommendation since 2008.

- Allows for query to have triple patterns, conjunctions, disjunctions and optional patterns.
SPARQL in perfect scenario

PREFIX rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#
PREFIX geo: http://www.geonames.org/ontology#
SELECT ?school
WHERE {
    ?state geo:featureClass geo:A.
    ?schools geo:featureClass geo:S.
    ?state geo:name “Ohio”.
}
Results

- Snapshot of retrieved results

<table>
<thead>
<tr>
<th>Expected Results</th>
<th>Actual Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wilber High School</td>
<td>Wilber High School</td>
</tr>
<tr>
<td>Cherokee Elementary School</td>
<td>Cherokee Elementary School</td>
</tr>
<tr>
<td>Buckeye High School</td>
<td>Buckeye High School</td>
</tr>
<tr>
<td>Middletown High School</td>
<td>Middletown High School</td>
</tr>
<tr>
<td>Fairborn Elementary School</td>
<td>Fairborn Elementary School</td>
</tr>
</tbody>
</table>

Since SPARQL works fine for perfect scenario, we do not need to evaluate other approaches for simple scenario.
SPARQL in not so perfect scenario

PREFIX rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#
PREFIX geo: <http://www.geonames.org/ontology#>

SELECT ?school
WHERE {
    ?state geo:featureClass geo:A.
    ?schools geo:featureClass geo:S.
    ?state geo:name “Ohio”.
}
Results

- Still works...

<table>
<thead>
<tr>
<th>Expected Results</th>
<th>Actual Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wilber High School</td>
<td>Wilber High School</td>
</tr>
<tr>
<td>Cherokee Elementary School</td>
<td>Cherokee Elementary School</td>
</tr>
<tr>
<td>Buckeye High School</td>
<td>Buckeye High School</td>
</tr>
<tr>
<td>Middletown High School</td>
<td>Middletown High School</td>
</tr>
<tr>
<td>Fairborn Elementary School</td>
<td>Fairborn Elementary School</td>
</tr>
</tbody>
</table>
SPARQL in final scenario...

PREFIX rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#
PREFIX geo: <http://www.geonames.org/ontology#>

SELECT ?school
WHERE {
    ?state geo:featureClass geo:A
    ?schools geo:featureClass geo:S.
    ?state geo:name "Ohio"
}

User has to know the exact structure and the precise relationships.
Path Expressions

- Finds paths in an RDF Graph given a source and a destination.
- Possible to specify constraints on the intermediate nodes e.g. path length, intermediate node, pattern constraint, ...
- Example: Find any feedback loops (i.e. non simple paths) that involve the compound Methionine.

```
SELECT ??p
    WHERE { ?x ??p ?x .
        ?z compound:name "Methionine" .
        PathFilter(containsAny(??p, ?z) ) }
```
Using path queries for slight and severe mismatch

- The semantics of the query changes to

  “Find schools related to Ohio”.

```
SELECT ??school
WHERE {
  ?state ??path ?school
  ?state geo:name “Ohio” .
  PathFilter( cost(??path) < 2 )
}
```

User has to know the path length for retrieving correct results.
• If available paths are

• Ohio has_County Greene County has_school Wilber High School

• Ohio has_County Montgomery County has_school Dayton School

• Ohio has_County Adams County has_school Buckeye High School

• Ohio has_County Lake County exchanges_student Nashville High

• Ohio has_County Greene County exchanges_student Seattle Aca.
## Results

- Snapshot of retrieved results

<table>
<thead>
<tr>
<th>Expected Results</th>
<th>Actual Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wilber High School</td>
<td>Wilber High School</td>
</tr>
<tr>
<td>Dayton School</td>
<td>Dayton School</td>
</tr>
<tr>
<td>Buckeye High School</td>
<td>Buckeye High School</td>
</tr>
<tr>
<td>Nashville High School</td>
<td>[Crossed out]</td>
</tr>
<tr>
<td>Seattle Academy</td>
<td>[Crossed out]</td>
</tr>
</tbody>
</table>
So where do these mechanism stand..

<table>
<thead>
<tr>
<th></th>
<th>Ease of writing</th>
<th>Expressivity</th>
<th>Works in all scenarios</th>
<th>Schema agnostic</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPARQL</td>
<td>X</td>
<td>√</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Path Expression</td>
<td>√</td>
<td>√</td>
<td>X</td>
<td>√</td>
</tr>
</tbody>
</table>
Proposed Approach
Proposed Approach

- Define operators to ease writing of expressive queries by implicit usage of semantic relations between query terms and hence remove the burden of expressing named relations in a query.

- Define transformation rules for operators based on work by Winston’s taxonomy of part-whole relations.

- Rule based approach allows applicability in different domains with appropriate modifications.
Architecture

SELECT ?school
WHERE { ?school geo:childrenFeature Ohio. }

Transformation Rules

• Triple Constraints
• Query Variables

Meta rules for Winston’s Categories

Mapping of ontology properties to Winston’s categories

User submits SPARQL Query

Query Rewriting Engine

Rewritten Query according to the data structure

SELECT ?school
WHERE { ?state geo:name "Ohio".
?county geo:childrenFeatures ?schools .}
Example Rules

• Transitivity
  - \((a \phi\text{-part of } b) (b \phi\text{-part of } c) \rightarrow (a \phi\text{-part of } c)\)
  - \((\text{Dayton place-part of Ohio}) (\text{Ohio place-part of US}) \rightarrow (\text{Dayton place-part of US})\)

• Overlap
  - \((a \text{ place-part of } b) (a \text{ place-part of } b) \rightarrow (b \text{ overlaps } c)\)
  - \((\text{Sri L. place-part of Indian Ocean}) (\text{Sri L. place-part of Bay of Bengal}) \rightarrow (\text{Indian Ocean overlaps with Bay of Bengal})\)

• Spatial Inclusion
  - \((a \text{ instance of } b) (c \text{ spatially included in } a) \rightarrow (c \text{ spatially included in } b)\)
  - \((\text{White House instance of Building}) (\text{Barack is in White House}) \rightarrow (\text{Barack is in building})\)
Perfect Scenario

SELECT ?school
WHERE {
  ?state geo:featureClass geo:A
  ?schools geo:featureClass geo:S.
  ?state geo:name "Ohio"
  ?schools in ?state
}

SELECT ?school
WHERE {
  ?state geo:featureClass geo:A
  ?schools geo:featureClass geo:S.
  ?state geo:name "Ohio"
  ?state geo:childrenFeatures ?schools
}
Slight and Severe Mismatch

SELECT ?school
WHERE {
  ?state geo:featureClass geo:A
  ?schools geo:featureClass geo:S.
  ?state geo:name "Ohio"
  ?schools in ?state .
}

---

SELECT ?school
WHERE {
  ?state geo:featureClass geo:A
  ?schools geo:featureClass geo:S.
  ?state geo:name "Ohio"
  ?county geo:childrenFeatures ?schools
}
So where do we stand with all these mechanisms..

<table>
<thead>
<tr>
<th></th>
<th>Ease of writing</th>
<th>Expressivity</th>
<th>Works in all scenarios</th>
<th>Schema agnostic</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPARQL</td>
<td>X</td>
<td>√</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Rho-Operator</td>
<td>√</td>
<td>√</td>
<td>X</td>
<td>√</td>
</tr>
<tr>
<td>Our Approach</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
</tbody>
</table>
Future Work
Evaluation

• Evaluate architecture on publicly available datasets such as Geonames, Sensor Ontology.

• Provide framework to execute schema agnostic complex queries such as
  – Find sensor systems to track blizzards in Ohio.
  – Find sensor systems to track blizzards in Ohio between Dec 25\textsuperscript{th}-27\textsuperscript{th} 2009.
  – .....
Conclusion

- Query engines expect user to know the structure of ontology and pose well formed queries.

- Query engines ignore semantic relations between query terms.

- Need to exploit semantic relations between concepts for processing queries.

- Need to provide systems to perform behind the scene rewrite of queries to remove burden of knowing structure of data from the user.
References

• SPARQL [http://www.w3.org/TR/rdf-sparql-query/](http://www.w3.org/TR/rdf-sparql-query/)

• Matthew Perry, Amit Sheth, Farshad Hakimpour, Prateek Jain “Supporting Complex Thematic, Spatial and Temporal Queries over Semantic Web Data”, Second International Conference on Geospatial Semantics (GEOS '07)

• Kemafor Anyanwu, Angela Maduko, and Amit P. Sheth, “SPARQ2L: Towards Support For Subgraph Extraction Queries in RDF Databases”, 16th international conference on World Wide Web (WWW’ 07)
Thank You!
Sensor Ontology

• Consists of roughly 90 classes and 33 object properties

• Data instantiated by querying Mesowest service.

• Exhibits meronymy between various concepts
  – Places, System, Compound Observation,....

  – Existing queries on dataset use Meronymy implicitly.
Existing Queries with Sensor Ontology

• Existing Queries for Sensor Ontology
  • Query a specific sensor for specific property
    • Example: “Find temperature recorded by System1”
  • Query a specific sensor for specific property with certain belief value. (Belief value assigned randomly as of now)
    • Example: “Find temperature recorded by System1 with belief value 0.73”
  • Query from a specific sensor system for a specific feature with a belief value
    • Example: “Find blizzards recorded by System1”
  • Query from a specific sensor system for a specific feature within a time interval
    • Example: “Find blizzards recorded by System1 during 1st Feb 1984-23rd Feb 1984”
Geonames Dataset

- Description at http://www.geonames.org/ontology/

- 100395794 (100 Million) RDF triples present in the dataset.

- Most interesting properties “parentFeature” (Administrative Region which contains the entity) and “nearbyFeature” (Entities close to this region).