SPARQL Query Re-writing Using Partonomy Based Transformation Rules*

Prateek Jain[1], Peter Z. Yeh[2], Kunal Verma[2],
Cory A. Henson[1], and Amit P. Sheth[1]

[1] Kno.e.sis, Computer Science Department, Wright State University,
Dayton, OH, USA
{prateek,cory,amit}@knoesis.org
[2] Accenture Technology Labs,
San Jose, CA, USA
{peter.z.yeh,k.verma}@accenture.com

Abstract. Often the information present in a spatial knowledge base is represented at a different level of granularity and abstraction than the query constraints. For querying ontology's containing spatial information, the precise relationships between spatial entities has to be specified in the basic graph pattern of SPARQL query which can result in long and complex queries. We present a novel approach to help users intuitively write SPARQL queries to query spatial data, rather than relying on knowledge of the ontology structure. Our framework re-writes queries, using transformation rules to exploit part-whole relations between geographical entities to address the mismatches between query constraints and knowledge base. Our experiments were performed on completely third party datasets and queries. Evaluations were performed on Geonames dataset using questions from National Geographic Bee serialized into SPARQL and British Administrative Geography Ontology using questions from a popular trivia website. These experiments demonstrate high precision in retrieval of results and ease in writing queries.

Keywords: Geospatial Semantic Web, Spatial Query Processing, SPARQL, Query Re-writing, Partonomy, Transformation Rules, Spatial information retrieval.

1 Introduction

Recently, spatial information has become widely available to consumers through a number of popular sites such as Google Maps, Yahoo Maps and Geonames.org [1]. In the context of the Semantic Web, Geonames has provided RDF [2] encoding of their knowledge base. One issue that makes using the Geonames ontology, or any non-trivial spatial ontology difficult to use, is that users have to completely understand the structure of the ontology before they can write meaningful queries. To illustrate our point, consider the following query from National Geographic Bee [3], "In which

* The evaluation components related to this work are available for download at http://knoesis.wright.edu/students/prateek/geos.htm

K. Janowicz, M. Raubal, and S. Levashkin (Eds.): GeoS 2009, LNCS 5892, pp. 140–158, 2009.

country is the city of Pamplona?" This seems to be a straightforward question, and one would assume that the logic for encoding this question into SPARQL [4] query would be to ask – Return a country which contains a city called Pamplona. However, it turns out that such a simple query does not work. This is because Pamplona is a city within a state, within the country of Spain. Therefore the correct logic for encoding the question into query would be – Return a country which contains a state, which contains a county, which contains a city called Pamplona. Unless the user fully understands the structure of the ontology, it is not possible to write such queries.

In this paper, we describe a system called PARQ (Partonomical Relationship Based Query Rewriting System) that will automatically align the gap between the constraints expressed in user's query and the actual structured representation of information in the ontology. We leverage existing work in classification of partonomic relationships[5] to re-write queries.

To study the accuracy of our approach for re-write, we tested it on (1) 120 randomly selected questions from the National Geographic Bee and evaluated them on Geonames ontology (2) 46 randomly selected trivia questions related to British villages and counties from trivia website[22] and evaluated them on British Administrative Geography Ontology[23]. For both the evaluations, users were instructed to read the questions and to write queries in SPARQL for the questions. PARQ rewrote the queries using partonomical relationships. The results were encouraging, and on an average, for evaluation 1, PARQ was able to re-write and answer 84 of 120 queries posed by users, whereas a SPARQL processing system could answer only 20 such queries. For evaluation 2, PARQ was able to re-write and answer 41 of 46 queries posed by users. For both the evaluations, we also compare the performance of PARQ with another well known system PSPARQL [24] which extends SPARQL with path expressions to allow use of regular expressions with variables in predicate position of SPARQL.

The contributions of this work are the following:

1. This work focuses on rewriting SPARQL Queries, written from a user's perspective without worrying about the underlying representation of information.
2. Our work utilizes partonomic transformation rules to re-write SPARQL queries.
3. PARQ has been completely evaluated on third party data (queries and dataset) and shows that it is able to re-write and answer queries not answered by a SPARQL processing system. We demonstrate PARQ can significantly improve precision without any recall loss.

The rest of the paper is organized as follows: section 2 discusses the background work, section 3 discusses approach followed by evaluation in Section 4. In Section 5, we discuss the related work and finally we conclude with section 6.

2 Background

All spatial entities are fundamentally part of some other spatial entity. Hence, spatial query processing systems often encounter queries such as (1) querying for parts of

spatial entities (for example, give me all counties in Ohio) (2) querying for wholes which encompass spatial parts (for example, return a country which contains a city called Pamplona).

By identifying which relationships between spatial entities are partonomic in nature it becomes feasible to identify if queries involving those relationships fail because of part-whole mismatch and it becomes possible to fix the mismatches using transformation rules that leverage the partonomic relationships. In this section, we will provide a brief overview of work related to partonomic relationships.

Our work of query rewriting to remove these mismatches is based upon using well-accepted partonomic relationships to address mismatches between a user's conceptualization of a domain and the actual information structure.

Part/Whole relation, or partonomy, is an important fundamental relationship which manifests itself across all physical entities such as human made objects (Cup-Handle), social groups (Jury-Jurors) and conceptual entities such as time intervals (5th hour of the day). Its frequent occurrence results in manifestation of part-for-whole mismatch and whole-for-part mismatch within many domains especially spatial datasets.

Winston [5] created a categorization of part whole relations which identified and covers part whole relations from a number of domains such as artifacts, geographical entities, food and liquids. We believe it is one of the most comprehensive categorization of partonomic relationships and other works in similar spirit such as [6] analyze his categorization.

This categorization has been created using three relational elements:

1. Functional/Non-Functional (F/NF):- Parts are in a specific spatial/temporal relation with respect to each other and to the whole to which they belong. Example: Belgium is a part of NATO partly because of its specific spatial position.
2. Homeomerous/Non-Homeomerous (H/NH):- Parts are same as each other and to the whole. Example: Slice of pie is same as other slices and the pie itself [5].
3. Separable/Inseparable (S/IN): - Parts are separable/ inseparable from the whole. Example: A card can be separated from the deck to which it belongs.

Table 1 illustrates these six different categories, their description using the relational elements and examples of partonomic relationships covered by them.

Using this classification and relational elements, relations between two entities can be marked as partonomic or non partonomic in nature. Further if they are partonomic, the category to which they belong is identified. Finally, appropriate transformation rules can be defined for each category to fix these mismatches.

For the purpose of this work, we have focused our attention on the last category "Place-Area". Places are not parts of any area because of any functional contribution to the whole, and they are similar to the other places in the area as well. Also places cannot be separated from the area to which they belong. Hence, this classification can allow appropriate ontological relationships to be mapped to Place-Area category such as those found in Geonames.

Table 1. Six type of partonomic relation with relational elements

Category	Description	Example
Component-Integral Object	Parts are functional, non-homeomerous and separable from the whole.	Handle-Cup
Member-Collection	Parts are non functional, non homeomerous and separable from the whole.	Tree-Forest
Portion-Mass	Parts are non functional, homeomerous and separable from the whole.	Slice-Pie
Stuff-Object	Parts are non functional, non-homeomerous and not separable from the whole.	Gin-Martini
Feature-Activity	Parts are functional, non-homeomerous and not separable from the whole.	Paying-Shopping
Place-Area	Parts are non functional, homeomerous and not separable from the whole.	Everglades-Florida

3 Approach

At the highest level of abstraction, PARQ takes in a SPARQL query and transforms it with the help of transformation rules. This section provides the details of our system. We describe the various modules of the system, the technologies used for building the system, the transformation rules utilized for transformation of the SPARQL queries and the motivation behind them. Finally we describe the underlying algorithm that explains how the transformation rules are utilized by PARQ for re-writing queries.

3.1 System Architecture

PARQ consists of following three major modules: 1) Mapping Repository 2) Transformation Rule generator and 3) Query Re-writer. Figure 1 illustrates the overall architecture of this system.

Mapping Repository. This module stores mappings of ontological properties to Winston's categories. These mappings are utilized by the Transformation Rule Generator to generate domain specific rules, which are consumed by the Query Re-writer. This is the only module in our system which requires user interaction (other than for query submission). In other words, the user has to specify these mappings.

Each mapping is encoded as a rule in Jena's rule engine format where the antecedent is a triple specifying an ontological property to be mapped and the consequent is a triple specifying the Winston category that the property is mapped to. For example, the following mapping:

[parentFeature: (?a geo:parentFeature ?b)=>(?a place_part_of ?b)]

maps “parentFeature” – a property from the Geonames ontology – to “place_part_of” – Winston’s category of Place-Area.

Transformation Rule Generator. This module automatically generates domain specific transformation rules using the mapping repository and pre-defined meta-level transformation rules based on Winston’s categories of part-whole relations, which we will explain later. For example, given the following meta-level transformation rule:

[transitivity_placePartOf: (?a place_part_of ?b)(?b place_part_of ?c)=>(?a place_part_of ?c)]

This module will utilize the parentFeature mapping defined above to generate the following domain specific transformation rule.

[transitivity_parentFeature: (?a geo:parentFeature ?b)(?b geo:parentFeature ?c)=>(?a geo:parentFeature ?c)]

The resulting rule is used by the Query Re-writer to re-write the graph pattern of SPARQL queries in the event of a partonomic mismatch.

This design enables PARQ to be easily used with a wide-range of ontologies. The knowledge engineer only needs to specify the mappings between properties of these ontologies and Winston’s categories, which requires less effort than generating the domain-specific transformation rules themselves. This design also allows the transformation rules to be extended in an ontology agnostic manner.

We implemented this module using Jena’s [7] rule engine API. Like the mappings, the meta-level transformation rules and the generated rules are encoded in the format accepted by Jena rule engine API. The rule engine allows reading, parsing and processing of rules along with the creation and serialization of new rules.

Query Re-writer. This module re-writes a SPARQL query in case of a partonomic mismatch between the query and the knowledge base to which the query is posed. This module is implemented using Jena and ARQ API [8]. Jena and ARQ provide functionality to convert a query into algebraic representation and vice versa. The triples specified in the query are identified. If they map to partonomic relation using the mapping repository and using Jena’s Rule Engine API, the domain specific transformation rule, appropriate transformation is performed on the triples. These transformations are then utilized to re-write the triples exhibiting the mismatch using the features provided by ARQ API.

We believe including transitivity as a part of the reasoner can result in significant overhead for large datasets such as geonames where transitivity applies to almost all the entities. By including it as a part of query rewriting method (1) it allows the mismatches to be resolved on an "on demand" basis (2) it makes it easy to plug in support for resolving other kinds of mismatches.

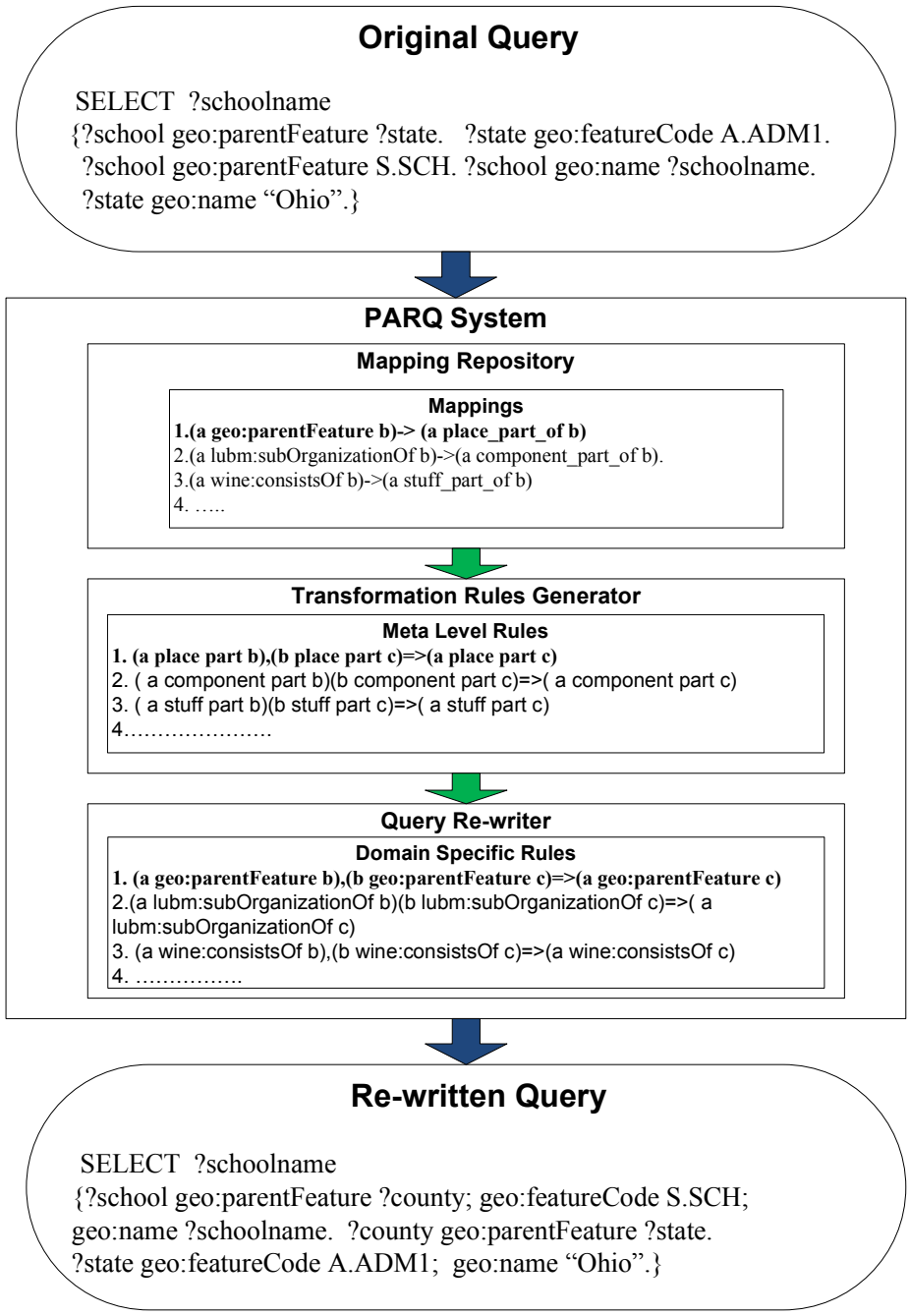

Fig. 1. PARQ System Architecture. The relevant rules and mappings for queries shown are highlighted in bold.

3.2 Meta-level Transformation Rules

Meta-level transformation rules are used to generate domain-specific rules that are used to resolve mismatches resulting from differences in encoding between the granularity of query constraints and the knowledge base by transforming the encoding of the constraints in the query to match the knowledge base.

These meta-level rules are defined at the level of Winston's categories, and a rule defined for a particular category applies to only the partonomic relations covered by that category. For example, rules defined for Component-Object category will cover only relations between machines and their parts, organization and their members, etc.

We used the following methodology to define the meta-level rules used by our system. First, we leveraged previous work by Varzi[9] and Winston, who both showed the semantics of transitivity holds true as long as it is applied across the same category of partonomic relation. From this result, we defined the meta-level transitive transformation rules shown in Table 2, that correspond to Winston's six part-whole categories.

Table 2. Transitivity for Winston's categories

ID	Antecedent1	Antecedent2	Consequent
1	a component part of b	b component part of c	A component part of c
2	a member part of b	b member part of c	A member part of c
3	a portion part of b	b portion part of c	A portion part of c
4	a stuff part of b	b stuff part of c	A stuff part of c
5	a feature part of b	b feature part of c	A feature part of c
6	a place part of b	b place part of c	A place part of c

Next, we investigated the interaction between Winston's categories by examining all possible combinations of these categories for additional transformation rules. This investigation, however, resulted in only frivolous rules, which were not useful for resolving mismatches. For example, the following transformation rule resulted from composing the Feature-Activity category with the Place-Area category.

(a place_part_of b) (b feature_part_of c) => (a feature_part_of c)

However given the following query and triples in an ontology (given in English for brevity),

QUERY: "What state was attacked in WW-II?"

TRIPLE 1 : Florida is a place part of USA (Place-Area).
TRIPLE 2: USA was attacked in WW-II (Feature-Activity)

The rule incorrectly transformed this query to match the ontology, that resulted in an incorrect answer being returned (i.e. Florida).

The reason for these frivolous rules is because Winston's categories are mutually exclusive as they are defined using relational elements. Hence, our meta-level transformations consist of only transitive rules. Despite this small number of rules, we found – through our evaluation – that transitivity by itself provide significant leverage in resolving part-whole mismatches.

3.3 Algorithm

The algorithm used in applying transitivity for resolving mismatches is as follows

SPR= Set of Partonomic Relation
If the query is not well formed
return
else
Convert the query Q into its algebric representation (AR).
Identify the graph pattern(GP) and query variables(QV).
For every triple t ∈ GP
if t.property ∈ SPR
If t.subject is a variable
Identify other triples with t.subject and use them to unify t.subject
Insert unified values in s.List
else
Insert t.subject in s.List
If t.object is a variable
Identify other triples with t.object and use them to unify t.object
Insert unified values in o.List
else
Insert t.object in o.List
for each s ∈ s.List
for each o ∈ o.List
path= Find path between s and o using the transformation rule.
If (path! =null)
Replace the resources in the path such that,
path.source = t.subject.
path.destination = t.object
The intermediate nodes are replaced such that the object and subject of contiguous triples have the variable names.
Replace the triple in the graph pattern with the path containing the variables.

Return the query Q' to the user

Explanation

Let us explain the algorithm using a query "In which county can you find the village of Crook that is full of lakes?" If the SPARQL Query submitted by user for this question is

```
SELECT  ?countyName
WHERE
 { ?village ord:hasVernacularName  "Crook" .
   ?county  rdf:type              ord:County ;
            ord:hasVernacularName  ?countyName ;
            ord:spatiallyContains  ?village .
 }
```

Step 1: The system compiles the query to verify if it is well formed. Since, in this case it is a well written query, the system moves on to Step 2.

Step 2: The query is converted into its algebraic representation, and the system iterates through its list of triples to identify triples containing partonomic relationship using the mapping file provided by the user. In this case the last triple

t=?county ord:spatiallyContains ?village

contains "spatiallyContains" property which indicates that the object is part of the subject. Hence, this triple is identified as a triple for re-writing.

Step 3: The other triples which contain the variables mentioned in "t",such as:

?village ord:hasVernacularName "Crook".,
?county rdf:type ord:County.
?county ord:hasVernacularName ?countyName.

are utilized for unifying the values of variables of t (i.e. ?village and ?county). Using these ?village ={ osr7000000000013015 } which is the resource for "Crook" in Administrative Geography Ontology and ?county={set of resources belonging to counties} is computed.

Step 4: The set of unified values from Step3 is then utilized to compute a path by executing transformation rule of transitivity involving the property "tangentiallySpatiallyContains", "completelySpatiallyContains"

?place ={osr7000000000013015} ?county={List of counties}.This results in the following path being returned:

1. osr7000000000013244 tangentiallySpatiallyContains osr7000000000012934
2. osr7000000000012934 completelySpatiallyContains osr7000000000013015

Step 5: In the path, the source and destination are replaced as mentioned in the original query, and the intermediate node is consistently replaced by a variable.

1. ?county ord:tangentiallySpatiallyContains ?var
2. ?var ord:completelySpatiallyContains ?village.

Step 6: In the original query the last triple is replaced by these two triples resulting in the following query

```
SELECT  ?countyName
WHERE
{
   ?village  ord:hasVernacularName  "Crook" .
   ?county  rdf:type  ord:County ;
              ord:hasVernacularName  ?countyName ;
              ord:tangentiallySpatiallyContains ?var .
    ?var       ord:completelySpatiallyContains  ?village .
}
```

There can be certain cases where a number of paths are computed between two end points because of transitivity. This will result in generation of multiple re-written queries. We try to rank these generated queries using the following parameters: (1) Re-written queries generating results are given higher ranking than ones which do not (2) If both queries generate results, in those scenarios queries requiring minimum amount of re-writing are given a higher ranking.

4 Evaluation

We present two evaluations to assess the performance of our approach on resolving partonomic mismatches between SPARQL queries written by users and the ontology's to which these queries are posed. We perform these evaluations using: (1) Questions from National Geographic Bee on Geonames Ontology (2) Questions from a popular trivia website which hosts quiz related to "British Villages and Counties" on British Administrative Geography Ontology.

4.1 Evaluation Objective and Setup

Our objective is to determine whether our approach enables users to successfully pose queries about partonomic information to ontology where the users are not familiar with its structure and organization. This lack of familiarity will result in many mismatches that need to be resolved in order to achieve good performance.

To evaluate our objective, we chose Geonames [1] and British Ordinance Survey-Administrative Geography Ontology [23] as our ontology's because: (1) they are one of the richest sources of partonomic information available to the semantic web community. (2) they are rich in spatial information. Geonames has over 8 million place names – such as countries, monument, cities, etc. – which are related to each other via partonomic relationships corresponding to Winston's category of Place-Area. For example, cities are parts of provinces and provinces are parts of countries. Table 3 shows some key relationships found in Geonames.

Table 3. Geonames important properties

Property	Description
http://www.geonames.org/ontology#name	Name of the place
http://www.geonames.org/ontology#featureCode	Identifies if the place is a country, city, capital etc.
http://www.geonames.org/ontology#parentFeature	Identifies that the place identified by domain is located within the place identified by the range

Similarly, Administrative Geography Ontology provides data related to location of villages, counties and cities of the United Kingdom which again map to Winston's place-area relation. Table 4 shows the description of key administrative geography ontology properties. Namespace has been omitted for brevity.

Table 4. Administrative Geography important properties

Property	Description
spatiallyContains	The interior and boundary of one region is completely contained in the interior of the other region, or the interior of one region is completely contained in the interior or the boundary of the other region and their boundaries intersect.
tangentiallySpatiallyContains	The interior of one region is completely contained in the interior or the boundary of the other region and their boundaries intersect. It is a sub-property of spatiallyContains.
completelySpatiallyContains	The interior and boundary of one region is completely contained in the interior of the other region. It is a sub-property of spatiallyContains.

For evaluating our approach on Geonames ontology, we constructed a corpus of queries for evaluation by randomly selecting 120 questions from previous editions of National Geographic Bee[3], an annual competition organized by the National Geographic Society which tests students from across the world on their knowledge of world geography. For British Administrative Geography ontology, we selected 46 questions from a popular trivia website [22] that hosts a number of quizzes related to British geography. We chose these questions for evaluation because:

- These questions are publicly available, so others can replicate our evaluation.
- Each question has a well-defined answer, which avoids ambiguity when grading the performance of our approach.
- These questions are of places and their partonomic relationship to each other. Hence, there is significant overlap with Geonames and Administrative Geography Ontology.

Examples of such questions include:

- The Gobi Desert is the main physical feature in the southern half of a country also known as the homeland of Genghis Khan. Name this country.
- In which English county, also known as "The Jurassic Coast" because of the many fossils to be found there, will you find the village of Beer Hackett?

Once the questions were selected, we employed 4 human respondents (computer science students at a local university) to encode the corresponding SPARQL query for each question. These respondents are familiar with SPARQL (familiarity ranged from intermediate to advanced) but are not familiar with Geonames or Administrative Geography Ontology. These two conditions meet our evaluation objective.

For the National Geographic Bee questions, each subject was given all 120 questions along with a description of the properties in the Geonames ontology. Each subject was then instructed to encode the SPARQL query for each question using these properties and classes.

For the trivia questions, we employed only one human respondent to encode the corresponding SPARQL query because of limitations in time and resources. This respondent was given all 46 questions along with a description of the properties in the administrative geography ontology.

These instructions, original queries, responses and our source code is available for download at http://knoesis.wright.edu/students/prateek/geos.htm

4.2 Geonames Results and Discussion

We compared our approach to PSPARQL and SPARQL. PSPARQL [24] extends SPARQL with path expressions to allow use of regular expressions with variables in predicate position of SPARQL. The regular expression patterns allowed in PSPARQL grammar can be constructed over the set of uris, blank nodes and variables. For example, the following query when posed to PSPARQL returns all cities connected to the capital of France by a plane or train.

```
Select ?City2
WHERE
{  ?City1 ex:capital ex:France .
   ?City1 (ex:plane | ex:train) ?City2 . }
```

We posed queries encoded by human respondents (see previous subsection) to SPARQL and PARQ. We graded the performance of each approach using the metrics of precision (i.e. the number of correct answers over the total number of answers given by an approach) and recall (i.e. the number of correct answers over the total number of answers for the queries). We said an approach correctly answered a query if its answer was the same as the answer provided by the National Geographic Bee.

Table 5 shows the result of this evaluation for PARQ and SPARQL. PARQ on an average correctly re-writes 84 queries of the 120 posed by users performing significantly better than SPARQL processing system across all respondents ($p < 0.01$ for the X2 test in each case). The low performance (61 queries by using PARQ and 19 by

Table 5. Comparison Re-written queries Vs original SPARQL queries

	System	# of queries answered	Precision	Recall
Respondent1	PARQ	82	100%	68.3%
	SPARQL	25	100%	20.83%
Respondent2	PARQ	93	100%	77.5%
	SPARQL	26	100%	21.6%
Respondent3	PARQ	61	100%	50.83%
	SPARQL	19	100%	15.83%
Respondent4	PARQ	103	100%	85.83%
	SPARQL	33	100%	27.5%

SPARQL) for respondent 3 can be attributed to this subject having the least familiarity with writing queries in SPARQL and writing improper SPARQL queries. The high performance (103 queries using PARQ and 33 using SPARQL) for respondent 4, can be attributed to this subject having the most experience with SPARQL. For each respondent, the difference of 120 and re-written queries is the number of queries not re-written using PARQ.

For this comparison, we also compared the execution time of PARQ to PSPARQL as shown in Table 6. Because of limitations in time and resources, we were able to employ only one respondent to encode the queries posed to PSPARQL. Hence, we selected Respondent 4 because this respondent has the most experience and familiarity with SPARQL.

Table 6. Comparison PSPARQL and PARQ for Respondent 4

System	Precision	Recall	Execution time/query in seconds
PARQ	100%	86.7%	0.3976
PSPARQL	6.414%	86.7%	37.59

Although PARQ and PSPARQL deliver the same recall (86.7%), we clearly illustrate that PARQ performs much better than PSPARQL in precision ($p<0.01$ for X2 test) because of retrieval of multiple answers by PSPARQL even when the particular resource was present only once in the ontology, thus exhibiting a flaw in the underlying algorithm or implementation. It also illustrates that PSPARQL takes almost 95% more time on, average in answering a query than PARQ ($p<0.05$ for 2-tailed pair-wise t-test).

These results shows that mismatches are common when posing queries to an ontology and that our approach can successfully resolve these mismatches which enabled more queries to be correctly answered.

For example, given the question:

"In which country is Grand Erg Oriental?"

Most of the subjects produced the following query.

```
PREFIX geo:<http://www.geonames.org/ontology#>
SELECT ?countryname
 WHERE
   {      ?country   geo:featureCode geo:A.PCLI.
                     geo:name ?countryname.
          ?place     geo:name "Grand Erg Oriental";
                     geo:parentFeature ?country.}
```

This query, however, failed to return any results when posed to Geonames because in Geonames "Grand Erg Oriental" is represented as a part of "Tunis al Janubiyah Wilayat" (a state) which is a part of "Tunisia" (a country). PARQ was able to re-write the original query to align with Geonames (see rewritten query below) which enabled the correct result to be retrieved (i.e. Tunisia).

```
PREFIX geo:<http://www.geonames.org/ontology#>
SELECT ?countryname
 WHERE
   {
          ?country geo:featureCode geo:A.PCLI;
                   geo:name ?countryname.
          ?place   geo:name "Grand Erg Oriental".
                   geo:parentFeature ?var.
          ?var     geo:parentFeature ?country.
   }
```

4.3 Administrative Geography Ontology Results and Discussion

For the questions related to British villages and counties, we also compared our approach to PSPARQL. We did not compare our approach to SPARQL because it delivered poor performance in the previous evaluation. Because of time and resource limitations, we were able to employ only one respondent to serialize trivia questions related to British Villages for PARQ and PSPARQL. Again, we selected Respondent 4 for this task because this respondent has the most experience and familiarity with SPARQL, The performance of each approach was graded using precision and recall, and we also compared the execution time of both approaches. We said an approach correctly answered a query if its answer was the same as the answer provided by the trivia website. As illustrated in Table 7 PSPARQL and PARQ perform equally well for recall, but PARQ has a much better precision than PSPARQL ($p<0.01$ for X2 test). It also illustrates PSPARQL on an average is 28 times slower than PARQ ($p<0.05$ for the 2-tailed pair-wise t-test).

Table 7. Comparison PSPARQL and PARQ for Respondent 4

System	Precision	Recall	Execution time/query in seconds
PARQ	100%	89.13%	0.099
PSPARQL	65.079%	89.13%	2.79

These results again illustrate the fact that part-for-whole and whole-for-part mismatches are common in spatial ontology's and PARQ helps resolve these mismatches allowing users to write queries without worrying about the structure of the ontology. As for example for the following trivia question "In which English county, also known as "The Jurassic Coast" because of the many fossils to be found there, will you find the village of Beer Hackett?".

The user poses the following SPARQL query for the question (Namespace omitted for brevity).

```
SELECT  ?countyName
WHERE
{ ?village  ord:hasVernacularName  "Beer Hackett" .
  ?county   rdf:type              ord:County ;
            ord:hasVernacularName  ?countyName ;
            ord:spatiallyContains  ?village .
 }
```

The above specified query will not fetch any results because (1) the instance data for Administrative Geography models information using two subproperties of spatiallyContains namely "tangentiallySpatiallyContains" and "completelySpatiallyContains". (2) Villages may or may not be directly part of counties and may contain additional administrative divisions in between.

Unfortunately the difference between "tangentiallySpatiallyContains" and "completelySpatiallyContains" is very subtle and makes it extremely difficult for a naïve user to correctly identify and use the property for querying the ontology, unless the user looks at the instance data and identifies the properties. However, the property "spatiallyContains" is a parent property of both "tangentiallySpatiallyContains" and "completelySpatiallyContains" and is perhaps the most intuitive property of the ontology which captures the semantics of both the properties and can be used by a user for posing queries. So when the above mentioned query is re-written by PARQ according to ontology as following, it retrieves the correct result of "Dorset".

```
SELECT  ?countyName
WHERE
 { ?village  ord:hasVernacularName  "Beer Hackett" .
  ?county  rdf:type  ord:County ;
           ord:hasVernacularName  ?countyName ;
           ord:tangentiallySpatiallyContains ?var .
  ?var     ord:completelySpatiallyContains  ?village .
 }
```

4.4 Summary of Results and Limitations

Based on our experiments performed we have demonstrated that PARQ significantly improves precision without any loss in recall and performs significantly faster as well over other systems.

Although our approach significantly improved performance over PSPARQL and SPARQL, there were several queries that it could not answer. Our analysis uncovered the following reasons:

- Several queries (e.g. those about political entities) could not be answered because of insufficient information in Geonames. Example of such queries includes "The Cayman Islands are a territory of which country?"
- Some queries required additional transformations beyond the ones we have identified. These transformations involve relations such as containment and overlap of entities which cannot be defined in terms of Winston's categories. Hence, we need to extend Winston's categories to handle these types of mismatches. Example of such queries includes "Which continent contains the largest number of landlocked countries?"
- Some questions required features, such as aggregate functions, that are not part of the standard SPARQL specification. Our current focus is to provide support for features which are part of standard SPARQL specification. Example of such queries includes "Not including Taiwan, how many provinces comprise China?"

5 Related Work

To the best of our knowledge this is the first work which tries to allow users to formulate SPARQL queries from their perspective without having to worry about the structure of the ontology. However, there are existing works related to RDF Query processing and retrieval of spatial information some of which we think are worth mentioning to highlight their salient features and distinguish our work from them.

The use of Semantic Web technologies for better retrieval of spatial information by incorporating data semantics and exploiting it during the search process was illustrated in [26]. Building upon the vision of [26], for retrieval of spatial information, in our previous work [10] we have defined operators to query spatial, temporal and thematic information from RDF datasets. Our approach for retrieval of spatial information in that work utilizes metric parameters such as geometric co-ordinates, radius, buffer for defining various operators. The operators enhance the standard spatial operators provided by Oracle Spatial and are implemented as supplemental to SPARQL. The reliance on metric parameters compliments our approach here which relies on utilization of named relationships.

Another interesting approach for querying spatial information using SPARQL[11] advocates re-modeling of ontology, than extending SPARQL for retrieval of information. Because of the emphasis on remodeling ontology than transformation of query, this work is obviously along a different dimension than our work. But the work

discusses shortcomings of SPARQL for querying spatial data and discusses some interesting query types which a language tailored for spatial querying should be able to handle and hence motivates us in our work. In [17] authors discuss a system for storing spatial and semantic web data efficiently without sacrificing query efficiency which in future can help us in supporting various other kinds of queries.

In [12] we have defined operators for identifying paths in RDF dataset given a source and destination. Using these operators it is possible to express constraints such as the length of the path, specifying a particular node to include in the paths etc. Our current work differs from these works since this work is not on identifying paths. Additionally, our system re-writes SPARQL queries and does not require specification of source and destinations for results to be retrieved. In [25][24] investigate incorporation of regular expressions in the predicate position of SPARQL queries. Though some of these works can be used for answering the queries they suffer from issues of poor precision and slower execution time as demonstrated through our evaluation. Query re-writing has been investigated in other research areas such as databases for yielding better execution plans, data integration and semantic data caching in client-server system [19]. In context of query languages for structured graph data models, [20][21] deal with queries that involve transitive or repetitive patterns of relations in context of databases.

There has been work in spatial query processing system for retrieval of information using partonomic relation such as in [13][18], but not in the context of SPARQL and not utilizing named relationships. These works rely on the use of metric relations such as radius, distance etc. [13] focus on creation of composite or higher order objects via the process of thematic and spatial abstraction.

The work which comes close to our approach is [14]. The work utilizes OWL-DL entailment rules for re-writing SPARQL to retrieve inference results. Unlike our approach where we alter the original graph pattern, the queries are altered by extending graph pattern using UNION construct of SPARQL. In the absence of an accessible implementation, it becomes difficult to compare our approach with the system.

Another work SPARQL-DL[15] incorporates the semantics of SPARQL in their DL reasoner and hence, is along a different dimension than our work.

Some other works on query rewriting are related to Query Optimization [16] , but in our work we are more concerned with retrieval of information from spatial datasets by harnessing partonomic relationships than its optimization.

6 Conclusion and Future Work

We have presented an approach for supporting SPARQL rewriting to allow users to write queries from their perspective without having to worry about the structure of the ontology. Our experiments have been completely performed on third party dataset and queries. Using our experimental results we have proven that our system re-writes these queries using transformation rules such as transitivity effectively and thus helps in resolving the mismatch between query constraints and underlying knowledge base while maintaining a high level of precision of results. Further we have demonstrated that PARQ is significantly faster and can improve precision without any loss to recall.

Our future research ideas include support to handle mismatches that cannot be handled by transitivity alone such as overlap, spatial inclusion. We are investigating support for more SPARQL constructs such as FILTER, OPTIONAL pattern. We are also further testing our approach for its applicability across domains. Limited tests performed show that our approach performs well across other domains of partonomic relations as well. A systematic comparison between resolving mismatches using query re-writing method viz-a-viz a reasoner is part of some of the future goals of this work.

Acknowledgments. This research is funded primarily by NSF Award#IIS-0842129, titled "III-SGER: Spatio-Temporal-Thematic Queries of Semantic Web Data: a Study of Expressivity and Efficiency" and secondarily by NSF ITR Award #071441, “Semantic Discovery: Discovering Complex Relationships in Semantic Web”.

References

1. Geonames, http://geonames.org
2. Resource Description Framework, http://www.w3.org/RDF/
3. National Geographic Bee, http://www.nationalgeographic.com/geobee/
4. Prud’hommeaux, E., Seaborne, A.: SPARQL Query Language for RDF (2008), http://www.w3.org/TR/rdf-sparql-query/
5. Winston, M.E., Chaffin, R., Herrmann, D.: A Taxonomy of Part-Whole Relations. Cognitive Science 11, 417–444 (1987)
6. Peter, G., Simone, P.: A conceptual theory of part-whole relations and its applications. Data & Knowledge Engineering 20, 305–3227 (1996)
7. Jena, http://jena.sourceforge.net/
8. ARQ, http://jena.sourceforge.net/ARQ/
9. Varzi, A.C.: A note on the transitivity of parthood. Applied Ontology 1, 141–146 (2006)
10. Perry, M., Sheth, A., Hakimpour, F., Jain, P.: Supporting Complex Thematic, Spatial and Temporal Queries over Semantic Web Data. In: Fonseca, F., Rodríguez, M.A., Levashkin, S. (eds.) GeoS 2007. LNCS, vol. 4853, pp. 228–246. Springer, Heidelberg (2007)
11. Kolas, D.: Supporting Spatial Semantics with SPARQL. Terra Cognita, Karlsruhe (2008)
12. Anyanwu, K., Maduko, A., Sheth, A.P.: SPARQ2L: Towards Support For Subgraph Extraction Queries in RDF Databases. In: 16th World Wide Web Conference (WWW 2007), Banff, Canada (2007)
13. Omair, C., William, A.M.: Utilising Partonomic Information in the Creation of Hierarchical Geographies. In: 10th ICA Workshop on Generalisation and Multiple Representation, Moscow, Russia,
14. Jing, Y., Jeong, D., Baik, D.-K.: SPARQL Graph Pattern Rewriting for OWL-DL Inference Query. In: Fourth International Conference on Networked Computing and Advanced Information Management, NCM 2008 (2008)
15. Sirin, E., Parsia, B.: SPARQL-DL: SPARQL Query for OWL-DL. In: OWLED 2007 Workshop on OWL: Experiences and Directions, Innsbruck, Austria (2007)
16. Hartig, O., Heese, R.: SPARQL Query Graph Model for Query Optimization. In: Franconi, E., Kifer, M., May, W. (eds.) ESWC 2007. LNCS, vol. 4519. Springer, Heidelberg (2007)
17. Kolas, D., Self, T.: Spatially-Augmented Knowledgebase. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 792–801. Springer, Heidelberg (2007)

18. Vogele, T., Hübner, S., Schuster, G.: BUSTER-an information broker for the semantic web. In: KI, vol. 17, p. 31 (2003)
19. Halevy, A.Y.: Answering queries using views: A survey. VLDB Journal (2001)
20. Cruz, I.F., Mendelzon, A.O., Wood, P.T.: A graphical query language supporting recursion. SIGMOD Record 16, 323–330 (1987)
21. Cruz, I.F., Mendelzon, A.O., Wood, P.T.: G+: Recursive Queries Without Recursion. In: Expert Database Conference (1988)
22. http://www.funtrivia.com/
23. Administrative Geography Ontology, http://www.ordnancesurvey.co.uk/oswebsite/ontology/AdministrativeGeography/v2.0/AdministrativeGeography.rdf
24. Alkhateeb, F., Baget, J.-F., Euzenat, J.: Extending SPARQL with regular expression patterns (for querying RDF). Web Semantics 7, 57–73 (2009)
25. Pérez, J., Arenas, M., Gutierrez, C.: nSPARQL: A Navigational Language for RDF. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 66–81. Springer, Heidelberg (2008)
26. Egenhofer, M.J.: Toward the semantic geospatial web. In: 10th ACM international symposium on Advances in geographic information systems. ACM, New York (2002)