The ρ Operator: Discovering and Ranking Associations on the Semantic Web

Kemafor Anyanwu and Amit Sheth
Large Scale Distributed Information Systems Lab
Department of Computer Science, University of Georgia
Athens, Georgia 30602
{anyanwu, amit}@cs.uga.edu

ABSTRACT

In this paper, we introduce an approach that supports querying for `Semantic Associations` on the Semantic Web. Semantic Associations capture complex relationships between entities involving sequences of predicates, and sets of predicate sequences that interact in complex ways. Detecting such associations is at the heart of many research and analytical activities that are crucial to applications in national security and business intelligence. This in combination with the improving ability to identify entities in documents as part of automatic semantic annotation, gives a very powerful capability for semantic analysis of large amounts of heterogeneous content.

The approach for supporting Semantic Associations discussed in this paper has four main facets. First, it generalizes these associations into three main classes based on their structural properties, allowing us to reason about them in a domain-independent manner. The second is the provision of an operator ρ for expressing queries about such associations. Third, it uses a graph data model for knowledge representation, allowing the semantic associations search techniques to be built upon the graph algorithms for paths, while integrating knowledge from the schema into the search process. The fourth facet is the use of a notion of context, which allows for restricting the search space and for context-driven ranking of results. Just as a Web search engine looks for relevant documents in the current Web, ρ can be seen as discovering and ranking complex relationships in the Semantic Web.

In this paper, we demonstrate the need for supporting such complex semantic relationships. We also give a formal basis to the notion of Semantic Associations and give a brief discussion on our overall approach for discovering and ranking them.

Keywords

Semantic Associations, Semantic Relationships, Semantic Querying, Complex Relationship Discover, RDF, Semantic Web, Graph Data Model.

This research is funded in part by NSF-ITR-IDM Award # 0219649 titled "Semantic Association Identification and Knowledge Discovery for National Security Applications."

1. INTRODUCTION

Contemporary search engines and query systems reason about the meaning of document content on the Web by considering the structure of documents and set of words that are contained within them. Such a limited understanding of content has undermined the effectiveness of traditional querying and search techniques, especially in the light of the overwhelming growth rate of the Web. However, the emerging vision of the "Semantic Web" [3], and the related earlier efforts for semantic interoperability including Semantic Information Brokering [13][9] promise to enable newer and better techniques for reasoning about the meaning of content. In the Semantic Web, "machine processable meaning" of content will be made explicit by enabling documents to be "meaningfully" marked-up with ontological terms. Thus, ontologies have been identified as a key technology in enabling the Semantic Web. They allow us to conceptualize a domain by capturing the entities and relationships in the domain. Establishing what relationships an entity participates in gives some insight into its meaning, because it allows us see where the entity fits in relative to the rest of the domain. It would therefore be desirable that semantic query systems provide flexible mechanisms for querying about relationships.

The current efforts to realize the Semantic Web build upon two W3C standards. The first is the XML [4], which is a specification that enables data exchange by providing a standard syntax for data representation. The second is a standard data model for expressing the structure and very limited semantics of Web content called the Resource Description Framework (RDF) [5]. Whereas the XML data model is a node labeled graph with no label on edges (i.e. no relationships), the RDF data model is a node and edge labeled graph, enabling the specification of relationships between entities. Consequently, query languages for the XML data model such as XQuery [6], provide limited support for querying about relationships because edges are not interpreted, while languages such as RQL [9], provide some support for such queries. For example, one may ask as a query to find all entities that participate in a given relationship, where the specification of the relationship may be exact one, or a pattern to be matched given as a path expression. While these mechanisms allow for a wide range of queries to be expressed, there are still some important

classes of queries that are not expressible in these languages. Much of the reason for this lies in the nature of the querying paradigm used by these languages, in which relationships between entities must be specified as part of the query. However, to be able to ask a query like "*how* are entity X and entity Y related?", a different paradigm needs to be supported. This is because in this kind of query, the relationship between X and Y is an unknown and cannot be specified as a part of a query, but instead should be the result of the query. Furthermore, the entities and relations should be within the same document otherwise search engines can not locate inter-document relations.

Another limitation of contemporary systems is the limited notion of a relationship that is supported. Typically, this is limited to a single n-ary predicate and its transitive closure, or those that can be inferred by the given inference rules. However, different applications require support for different kinds of relationships, especially for applications in analytical domains such as national security and business intelligence. For example, in the light of the recent breach of flight security, it has become pertinent that airport security agents be able to ask questions like, "What *important relationships* exist between Passenger X and Passenger Y?" The notions of an important relationship in the context of assessing the risk of flight based on passenger associations, may involve capturing any link between two passengers that may be considered relevant at that time. For example, the fact that two passengers attended the same flight training school or that they had their tickets purchased using the same credit card, even though they do not have a known family or business relationship, may be an irrelevant association in many contexts. On the other hand, such a relationship may be considered very important in the context of the flight security. An association may be ascribed even more importance if additional associations were present like the fact that both passengers were citizens of, or had other associations to, countries that are considered terrorists states, terrorism sponsoring states, or terrorist harboring states. Note that it would not be possible to encode all of such associations as inference rules for every domain, and most times these associations are not known a priori, but are discovered with experience. Therefore, it is important to develop methods to identify or discover associations using general characteristics that are domain independent, and then apply domain knowledge to guide the search process, allowing the search to focus on only associations that are important in that context. Typically, these kinds of relationships are indicated by some kind of connections, patterns or similarities between entities. We call such complex relationships, `Semantic Associations`.

In this paper, we introduce a precise notion of a Semantic Association, along with an approach that supports their discovery in RDF knowledge bases. The approach is based on the use of a query operator ρ (Rho). The ρ operator exploits both important graph properties as well as schema knowledge, to find Semantic Associations between entities. Another important component of our approach is the use of context for delimiting search space and ranking results. Our notion of context captures the smallest possible scope for a user's query, as well as information like which relations are particularly important for a given context, allowing for greatly reducing the search space and a context sensitive ranking of results.

The rest of the paper is organized as follows: First, we will motivate our work by drawing a contrast with the approach taken by traditional query languages, and then illustrate the notion of Semantic Associations using an example. In section 3, we provide a formal basis for representing Semantic Associations. In Section 4, we give a brief discussion about our approach to compute ρ. Section 5 reviews some related, work and then section 6 concludes the paper.

2. MOTIVATION

We will now illustrate Semantic Associations by way of a simple example shown in Figure 1, which is a modification of the example used in [9]. Our modification introduces some contrived nodes and edges so as demonstrate several examples of Semantic Associations. The figure shows an RDF knowledge base containing information for a cultural portal from two perspectives reflected in two different schemas (the top part of the figure). One perspective captures museum specialist's perspective using concepts like `Museum`, `Artist`, `Artifact`, etc. (top left schema). The second perspective describes resources from a Portal administrator's perspective using concepts like `file-size`, `mime-type`, etc. (top right schema).

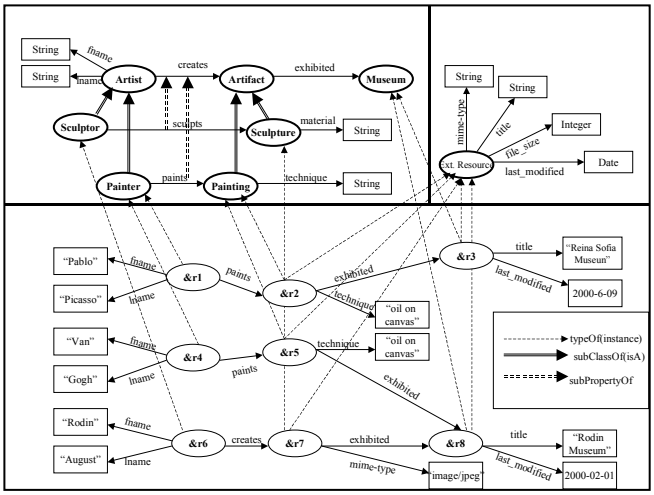

Figure 1: Example knowledge base

The lower part of the figure has descriptions about resources, e.g., museum websites (&r3, &r8), images of artifacts (&r2, &r5, &r7) and for resources that are not directly present on the Web, e.g., people, nodes

representing electronic surrogates are created (&r1, &r4, &r6).

The resources are connected by an arc if there is an rdf:property relating them. For example, the fact &r1 is connected to &r2 by a paints property means that resource represented by &r1 painted the resource &r2. In addition, resources are connected to resource classes using an rdf:typeOf property, which indicates that the resource is a member of the class it is connected to. For example, &r1 is a Painter. Furthermore, resources can belong to multiple unrelated classes. For example, &r3 and &r8 both belong to the Museum class in the left schema, and the Ext. Resource class in the right schema. In the schemas, classes and properties may also be related by special properties (rdfs:subclassOf, rdfs:subpropertyOf respectively), indicating a hierarchy on classes and properties.

To find semantic associations we look beyond single properties linking entities, by so doing we may find the following interesting associations:

i. &r1 and &r3 have an association because &r1 paints a Painting (&r2) which is exhibited at the museum (&r3)
ii. &r4 and &r6 are semantically associated because they both have created artifacts (&r5, and &r7) which are exhibited at the same museum (&r8).
iii. &r1 and &r6 are associated because of a similarity in their relationships. For example, they both have creations (&r2, and &r7) that are exhibited by a Museum (&r3, &r8).

The first two associations capture some connectivity between entities, while the last association captures a similarity between entities. Note that the notion of similarity used here is not just a structural isomorphism, but path semantic isomorphism of paths, which captures a semantic similarity of nodes and arcs along a path. In this notion of isomorphism, two entities are considered similar, if they are related by subclass relationship, or have a common ancestor class. For example in the third association, although one case involves a painting and the other a sculpture, we consider them similar because sculpting and painting are both kinds of creative activities, and Sculptures and Paintings are kinds of Artifacts. The notion of similarity is extended to properties.

Most contemporary query languages and systems allow us to query about functional relationships (i.e. attributes) of a given class of entities specified as a restriction clause, e.g., WHERE clause. Sometimes, the description includes a specification of some relationship that the qualifying entities have with other entities, e.g. JOIN condition. We call this paradigm the Query-for-Attribute paradigm. In this paradigm, a query is a specification of entities to be matched and the result of a query is the set of entities that match the criteria. The limitation of this paradigm is that relationships that exist between entities in the world must be known a priori, in order to specify them as a part of the query (i.e., the JOIN condition). Such a requirement may be reasonable in a single application environment where users are very conversant of the schemas used to represent data. However, in a distributed and heterogeneous environment such as the (Semantic) Web, such a requirement would not be reasonable, because there would be many different ontologies and schemas designed by people other than the users.

In order to support the kinds of queries that we mentioned earlier, query systems need to support another paradigm, which we call the Query-For-Relationship paradigm. In this paradigm, a query is given by the set of entities that are being studied, and the result of the query is the set of relationships that exist between them. The relationships may be as simple as individual n-ary predicates or as complex as semantic associations. This paradigm eliminates the requirement for a query to specify information about the relationship connecting two entities, and is necessary to support analytical tasks where, in fact, finding such a relationship is focus of the task.

What we need from semantic querying systems is the ability to find these kinds of associations for a given set of entities without any requirement on the user to provide information about the association as is done in the Querying-for-Attribute paradigm.

3. FRAMEWORK

Our framework is based on using a graph model to represent data. This approach offers us two main advantages: First, graph representations can capture in a natural way the connectivity between the entities. This enables us to map queries about connectivity to typical graph operations like path searches. Second, a wide range of information can be modeled as graphs including E-R models and RDF data models. In the sequel, we will summarize a graph data model that is largely based on the RDF data model.

3.1 Data Model

The RDF data model is a directed labeled graph, in which nodes are Resources (i.e., entities with URIs) or Literals, and edges are Properties. Properties are binary relations between entities. They are defined by a domain (i.e. the set of classes that they apply to) and a range (i.e. class of entities from which it takes its values). An RDF statement is a triple (Subject, Property, Object) asserting that the resource that is the subject of the statement has a property whose value is the object of the statement. An object can be either another resource or a literal value. A statement can be represented

as a graph, by drawing an arc from a node representing the subject to another node representing the object, `Subject` → `Object`. The arc is labeled with the name of the property and resource nodes are labeled with the URIs of the resources. A special property `rdf:typeOf` links resources to the classes they belong to. The classes and properties are described in an RDFS (RDF Schema) [5]. RDFS provides a general schema (classes and properties) for describing domain specific vocabularies. We can view an RDF Schema RS as a tuple (SC, SP) where SC is the set of classes in the schema and SP is the set of properties. Both SC and SP may be organized as hierarchies with subclass and sub-property relationships respectively. Following the terminology of [9], we call the set triples that conform to an RDF Schema its description base. An interpretation I for a schema RS, RS^I interprets classes in a schema as unary relations, i.e. for a class C, C^I is the set of resources that are its members in that interpretation. Properties are interpreted as binary relations, i.e., for a property P, P^I is a set of ordered tuples (a, b) where `a` is an element in C_i^I, where is the interpretation of f one of the domain classes under I, and `b` is an element in the interpretation of the range class under I.

3.2 Semantic Associations

In our framework, a query is a pair of entity identifiers for two resource entities (e1, e2). For example, in the RDF model it would be a pair of resource URIs. The result of a query is a set of Semantic Associations between entities. We will give a formal basis to the notion of a Semantic Association.

3.2.1 Definition 1 (Property Sequence)

A `Property Sequence` S is a finite sequence of properties $P_1, P_2, P_3, \ldots P_n$ where each P_i is a property defined in a RDF Schema R_j. The interpretation of a property sequence S is the set of ordered sequences of tuples:

$$S^I = \{[(a_1, b_1), (a_2, b_2) \ldots (a_n, b_n)] \mid (a_i, b_i) \text{ is in } P_i^I]\}$$

3.2.2 Definition 2 (Satisfiability)

A property sequence S is `satisfiable` if there exists a an interpretation I which satisfies it: i.e.

$$\forall P_i \exists (X_i, Y_i) \text{ in } P_i^I \mid Y_i = X_{i+1} \text{ for } i > 0 < n$$

This means that there is a sequence of tuples of the form (<a,b> <b,c> <c,d> ….<m,n>), where the i^{th} element of the sequence is an element in the interpretation of P_i under I. We use, S[i] to denote the i^{th} tuple element of the sequence S.

The node *a* is called the `origin` of the sequence and *e* is the `terminus`. The function `NodesOfPS()` returns the ordered set of nodes in a property sequence S, so that

`NodesOfPS(S)` = [a, b, c, d,…m, n]

3.2.3 Definition 2 (Joined Property Sequences):

A set of property sequences $S_1, S_2, S_3 \ldots S_n$ are called `joined` if are they satisfiable in an interpretation I and:

$$\bigcap \text{NodesOfPS}(S_i) \neq 0 \text{ for } i = 1..n$$

i.e. the sequences $S_1, S_2, S_3 \ldots S_n$ intersect at some node n in I.

3.2.4 Definition 3 (ρ-Isomorphic Property Sequences)

Two property sequences S1 = $P_1, P_2, P_3, \ldots P_m$, and S2 = $Q_1, Q_2, Q_3, \ldots Q_m$ are called `ρ-isomorphic` if:

1) both S1 and S2 are satisfiable in an interpretation I
2) $S_1[i] \cong_\rho S_2[i]$ for i = 1..m

where $P_x \cong_\rho Q_x$ implies the following conditions:

1. $\exists$ R, R is a property | P_x , $Q_x \subseteq$ R. This captures a parent-child relationship, a sibling relationship or equality between P_x and Q_x.
2. Similarly for the nodes connected by P_x and Q_x. For (a, b) in P_x^I, and (u, v) in Q_x^I , $\exists$ C_1, C_2 , C_3, , C_1, C_2 , C_3, are classes such that
   a. a $\in C_1$, u $\in C_2$ and $C_1, C_2 \subseteq C_3$ and
   b. b and v satisfy the same conditions.

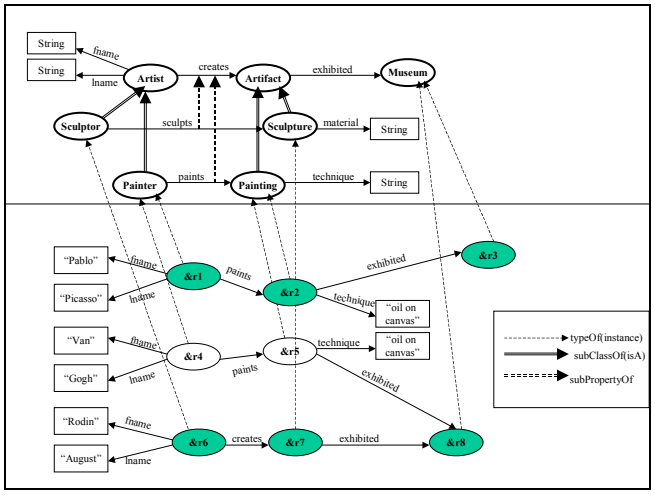

Figure 2 : Isomorphic Property Sequences

The second definition ensures that that the two paths maintain a similarity of both nodes and edges along the sequence. For example, in Figure 2, although &r1 is a painter and $r6 is a sculptor both are similar because they are both Artists. Also, because `paints` is a subproperty of `creates`, we consider them isomorphic. Therefore the sequences [*paints*, *exhibited*] with origin &r1, and

[*creates*, *exhibited*] originating at &r6 are ρ-isomorphic, making r1 and r6 semantically associated.

3.2.5 Definition 4 (Semantic Associations)

We now define Semantic Associations. . We say that two entities x and y are *semantically associated* in an instance base I if:

i. there is a property sequence S which is satisfiable in an instance base I such that either x is the origin and y is the terminus, or y is the origin and x is the terminus,

ii. there exists two joined property sequences S_1 and S_2 which are satisfiable in an instance base I such that x is the origin of S_1 and y is the origin S_2, or

iii. there exists two satisfiable ρ-isomorphic property sequences S_1, S_2 such that x is the origin of S_1 and y is the origin of S_2.

3.2.6 Definition 5 (SAQ – Semantic Association Query)

A Semantic Association Query – SAQ, is a pair of entity keys or ids (in the case of RDF it would be their URIs). The result of an SAQ is a set of Semantic Associations that exist between the entities identified in the query.

4. APPROACH

The ρ operator reasons about semantic association based on two main capabilities. First, it uses schema knowledge to decide whether or not an association is possible. If an association is not possible, the actual search is not done. For example, in searching for a similarity between entities, it first searches to see if both belong to a common class, a common parent class, or to classes that are related by a subclass relationship. If they do not, they it would not be possible to find ρ−isomorphic sequences as we have defined it, therefore the search is canceled. If a semantic association seems possible then a traversal is done in the description base.

Note that the traversal need not search all possible paths because the schema gives us an idea of potential paths. Another important component is the use of `context`. Our notion of context captures user specified scope declarations for limiting the search and optional user specified relevance ranking for properties or sequences. The scope declaration may be a restriction to a specific schema/s or a subset of a schema, thereby reducing the search space. In the above example, a user can decide to focus the search to the museum specialist's perspective. We then ignore the arcs and nodes not relevant in that context, as shown in the example in Figure 3, where the nodes and arcs not relevant in that schema are dropped. For large schemas, the restriction may be to a region of the schema. In addition, a user may specify some relevance rankings to properties, indicating which ones are most important, and if several, the order of importance. This information is used in ranking, thereby presenting those associations with the total highest relevance ranking first. Other heuristics are also employed for ranking. For example, in some contexts, longest sequences may be more interesting, while in other contexts, the shortest sequences will be used.

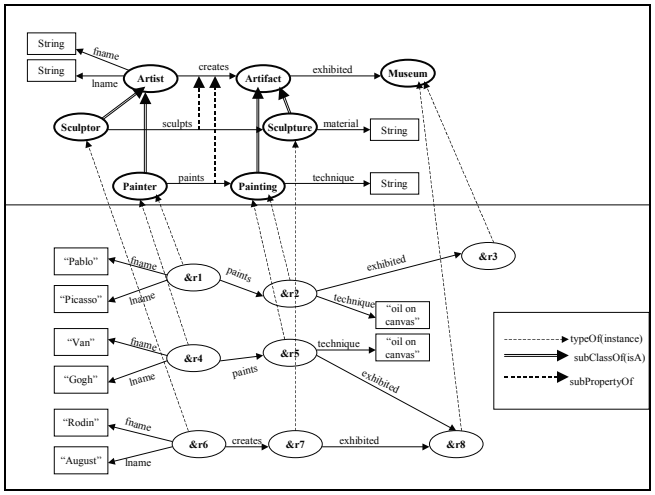

Figure 3 : Context Pruned Knowledge Base

5. RELATED WORK

As mentioned before, the pervasive query paradigm in contemporary query languages is the `Querying-for-Attribute` paradigm. Here, information about relationships between entities is specified as part of the query, i.e. join condition, which is inadequate for many applications, especially the upcoming semantic applications which exploit relationships as the core of its computation [16][17]. The ability to return as a result of query, associations between entities, which includes a path of relations between entities as well as other associations like similarities, is needed.

There was some earlier work in finding paths between entities through the use of recursive queries [1]. Recursive queries computed the transitive closure of a single predicate relation. However, these approaches still use the Querying-for-Attribute paradigm since the attribute has to be specified as part of the query. Similarly, in first order logic based systems, e.g. Datalog, Prolog, computation of paths require that the derivation rules be specified, which as earlier emphasized may not be a reasonable requirement for many applications. In fact, to specify these queries we would need to use second order rules or queries since it requires the use of the predicate variables. Moreover, the Semantic Association that identifies a similarity between entities may not be easily specified in these systems.

There is one effort however, that bears some resemblance to our work which is that of DISCOVER [8]. Their approach searches for associations in relational databases

that are similar to our notion of Joined Property Sequences, by exploiting primary key to foreign relationships specified in a schema. However, the meaning of the sequences found, have to be interpreted by users because a relational database schema contains limited information on data semantics. [18] also discusses an approach for semantic ranking of results which is based on estimating the distance between concepts in the query and concepts in the result, and assigning more relevance to results using closer concepts to query concepts. However, our work relates to ranking of contextually relevant complex relationships and hence the semantic distance (i.e., the measure used for ranking) has very different interpretation.

6. CONCLUSION

Query languages for the Semantic Web need to support additional query paradigms than those currently supported in order to meet the demands of novel applications in national security and business intelligence. There is also a need to support different notions of relationships varying from simple to complex. We have shown some examples of these complex relationships and presented an approach to computing them. Relationships are fundamental to semantics, and here we take one step towards discovering some types of semantic relationships.

7. ACKNOWLEDGEMENTS

We would like to acknowledge Dr. I. Budak Arpinar and Dr. John Miller for their insightful comments and suggestions.

8. REFERENCES

[1] R. Agrawal. Alpha: An Extension of Relational Algebra to Express a Class of Recursive Queries. *IEEE Transactions on Software Engineering*. 14(7): 879--885, July 1988.

[2] S. Alexaki, G. Karvounarakis, V. Christophides, D. Plexousakis and K. Tolle. The ICS-FORTH RDFSuite: Managing Voluminous RDF Description Bases. In *2nd International Workshop on the Semantic Web*, pages 1--13, Hong Kong, 2001.

[3] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, May 2001.

[4] T. Bray, J. Paoli, and C.M. Sperberg-McQueen. Extensible Markup Language (XML) 1.0. W3C Recommendation, February 1998.

[5] D. Brickley and R.V. Guha. Resource Description Framework (RDF) Schema Specification 1.0, W3C Candidate Recommendation. 2000.

[6] D. Chamberlin, D. Florescu, J. Robie, J. Simeon and M. Stefanescu. XQuery: A Query Language for XML. Working draft, World Wide Web Consortium, June 2001.

[7] R. H. Guting. GraphDB: Modeling and querying graphs in databases. *In Proceedings of the International Conference on Very Large Data Bases*, pp. 297--308, 1994.

[8] V. Hristidis and Y. Papakonstantinou. DISCOVER: Keyword Search in Relational Databases *In Proceedings of the International Conference on Very Large Data Bases* (VLDB 2002), Hong Kong, China 2002.

[9] G. Karvounarakis, S. Alexaki, V. Christophides, D. Plexousakis and M. Scholl, RQL: A Declarative Query Language for RDF, *WWW2002*, May 7-11, 2002, Honolulu, Hawaii, USA.

[10] V. Kashyap and A. Sheth. Semantics-Based Information Brokering. CIKM: 363-370, 1994.

[11] O. Lassila and R. Swick. Resource Description Framework (RDF) Model and Syntax Specification, W3C Recommendation. 1999.

[12] M. Mannino and L. Shapiro. Extensions to Query Languages for Graph Traversal Problems. *TKDE 2*(3): 353--363,1990 O. Mendelzon, P. Wood. Finding Regular Simple Paths in Graph Databases. *SIAM J. Comput*., 24(6):1235-1258, 1995.

[13] E. Mena, V. Kashyap, A. Illarramendi, and A. Sheth. Domain Specific Ontologies for Semantic Information Brokering on the Global Information Infrastructure. In International Conference on Formal Ontologies in Information Systems (FOIS'98), Trento (Italy), pages 269--283, 1998.

[14] O. Mendelzon, and P. Wood. Finding Regular Simple Paths in Graph Databases. *SIAM J. Comput*., 24(6):1235-1258, 1995

[15] R. Tarjan, Fast Algorithms for Solving Path Problems. *J. ACM Vol. 28, No. 3*, 594-614, July 1981.

[16] A. Sheth, C. Bertram, D. Avant, B. Hammond, K. Kochut, Y.Warke: Managing Semantic Content for the Web. IEEE Internet Computing 6(4): 80-87 (2002).

[17] A. Sheth, I.B. Arpinar and V. Kashyap, "Relationships at the Heart of Semantic Web: Modeling, Discovering, and Exploiting Complex Semantic Relationships, Enhancing the Power of the Internet: Studies in Fuzziness and Soft Computing, Masoud Nikravesh, Ben Azvin, Ronal Yager and Lotfi A. Zadeh, Eds., Springer-Verlag, 2003 (to appear).

[18] N. Stojanovic, A. Mädche, S. Staab, R. Studer, Y. Sure. SEAL -- A Framework for Developing SEmantic PortALs. In: K-CAP 2001 – Proceedings of the First International ACM Conference on Knowledge Capture, October 21-23, 2001, Victoria, B.C., Canada.