SemRank: Ranking Complex Relationship Search Results on the Semantic Web

Kemafor Anyanwu
LSDIS Lab
Department of Computer
Science
University of Georgia
Athens, Georgia, USA

anyanwu@cs.uga.edu

Angela Maduko
LSDIS Lab
Department of Computer
Science
University of Georgia
Athens, Georgia, USA

maduko@cs.uga.edu

Amit Sheth
LSDIS Lab
Department of Computer
Science
University of Georgia
Athens, Georgia, USA

amit@cs.uga.edu

ABSTRACT

While the idea that querying mechanisms for complex relationships (otherwise known as Semantic Associations) should be integral to Semantic Web search technologies has recently gained some ground, the issue of how search results will be ranked remains largely unaddressed. Since it is expected that the number of relationships between entities in a knowledge base will be much larger than the number of entities themselves, the likelihood that Semantic Association searches would result in an overwhelming number of results for users is increased, therefore elevating the need for appropriate ranking schemes. Furthermore, it is unlikely that ranking schemes for ranking entities (documents, resources, etc.) may be applied to complex structures such as Semantic Associations.

In this paper, we present an approach that ranks results based on how predictable a result might be for users. It is based on a relevance model SemRank, which is a rich blend of semantic and information-theoretic techniques with heuristics that supports the novel idea of modulative searches, where users may vary their search modes to effect changes in the ordering of results depending on their need. We also present the infrastructure used in the SSARK system to support the computation of SemRank values for resulting Semantic Associations and their ordering.

Categories and Subject Descriptors

H.3.3 [**Information Systems**] Information Search and Retrieval

General Terms

Algorithms, Experimentation, Measurement

Keywords

Semantic Web, SemRank, Semantic Ranking, Ranking Complex Relationships, Semantic Associations Search, Semantic Relationship Search, Semantic Match, Semantic Similarity, Discovery Query, Path Expression Tree, Semantic Summary

1 INTRODUCTION

The premise of search technologies today is primarily centered around enabling search for entities or on the Semantic Web resources. However, the following quote by Grady Booch [5] summarizes the limitations of a purely entity-centric world view:

"*An object by itself is intensely uninteresting*".

Similar sentiments were echoed in [23] where relationships were emphasized as the heart of the Semantic Web. Correspondingly, efforts must be made to extend or identify alternatives to traditional search mechanisms focused on finding documents described either by keywords or semantic annotations, with capabilities for searching about complex relationships between Semantic Web resources. Such search capabilities may become the foundation for a "Relationship Search Engine", a technology with the potential for immense real world value in analytical and knowledge discovery applications [27]. Relationship search technologies will provides effective means to answers questions such as "Does a semantic relationship exist between X and Y"? One step in this direction is [1][3] where the notion of Semantic Associations is formalized and refers to complex relationships between resources capturing the connectivity or similarity of the resources. In graph theoretic terms, they refer to labeled paths (not necessarily directed) that either connect resources or that the resources have similarly. In addition, progress [4][18]**Error! Reference source not found.** is now being made in developing efficient evaluation strategies for discovering such relationships on the Semantic Web.

Another important related issue that must be addressed by relationship search technologies is how to determine the relative importance of relationships found with respect to a user's context because that impacts how they will be ranked. This problem is particularly important because it is possible, in fact likely, that the number of such relationships are much larger than the number of entities themselves. This means that there is potential for creating a more acute information overload problem than currently exists on the Web. It is therefore imperative that we develop techniques for ordering search results in order to present results of highest importance first. Unfortunately, it is not clear that the techniques used currently for ranking entities on the Web e.g. for HTML, PageRank [7], HITS [14], for XML [13] [8], and on the Semantic Web, [23], can be used to rank complex relationships. There is some related work [26] being done in the area of ranking Semantic Networks. However, this approach suffers from the same limitation as most ranking approaches which have a fixed ranking scheme that imposes a single type of ordering on results. That is that the same query made in different contexts and for different

purposes, still yields the same ordering. It seems that some flexibility should be built into the relevance models so that different orderings may be imposed on the same result set depending on the user's need. For example, in an investigative context the focus of a search may be to uncover obscure relationships between entities (e.g., it is suspected that in money laundering, innocuous dealings/relationships are purposefully introduced to further obscure relationships [27]), whereas the focus of a conventional search may be to find predictable or commonly expected relationships for the purpose of validating or augmenting already known information. Therefore, in the earlier scenario, we may need to boost results that are considered unpredictable whereas in the latter case we may need to reverse that ordering.

The challenge here of course is in defining the metrics that will be used for determining the ordering of results. When using IR style techniques over structured or semi-structured data, the ordering of results is based on how close a document is to a query (i.e., an explicit description of a user's need). In context of a relationship search however, queries do not contain a description as such, they may just identify the entities of interest. Consequently, a relevance model that is based on how good of a match a document is to a query does not apply and the development of novel techniques is necessary. One promising approach to dealing with this problem is based on using metrics that somehow measure the predictability of the result that is being returned. For example, we may choose to rank highest in an investigative or discovery search, results that are less predictable while in a conventional search the reverse ordering more desirable.

1.1 Outline and Contributions

- In this paper, we focus on ranking the results of complex relationship searches on the Semantic Web. We pursue an approach that is based on a `modulative` relevance model SemRank, that can easily (using a sliding bar) be `modulated` or adjusted via the query interface. In this way, a user can easily vary their search mode from a `Conventional` search mode to a `Discovery` search mode based on their need. The richness of the SemRank relevance model stems from the fact that it uses a blend of semantic and information theoretic techniques along with heuristics to determine the rank of Semantic Associations. It also has the advantage of being a unified model for ranking all the different types of Semantic Associations.

- We discuss the infrastructure provided by the SSARK (*Semantic Searching of A different Kind*) system for the computation of SemRank values and ordering of results based on these values. Some key components of the infrastructure include the idea of `modulative` searches used to support anywhere from conventional to discovery searches; the notion of `Semantic Summaries` analogous to the notion of structural summaries used for optimizing path expression query evaluation; a pipelined Top-K algorithm for computing approximately correct orderings of search results.

The rest of the paper is organized as follows: section 2 presents some background information on Semantic Associations, section 3 discusses the issue of ranking Semantic Associations and presents the components of the SemRank model. Section 4 discusses the computational issues with respect to computing SemRank values and result ordering and presents the strategies adopted by the SSARK system. Section 5 presents an empirical evaluation of our approach and sections 6 and 7 discuss related work and conclusion respectively.

2 BACKGROUND AND MOTIVATION

Semantic Associations were based on the notion of RDF `Property Sequences` whose instances can be viewed as labeled paths in a knowledge base. A knowledge base in our context refers to a set of RDF descriptions or OWL-Lite descriptions represented in RDF. They also include certain binary relations on Property Sequences that capture the intersection and similarity of paths in which entities are involved. Figure 1 shows some examples of Semantic Associations. Figure 1(a) shows a direct path connecting resources r1 and r2 and is called a `ρ-pathAssociation`.
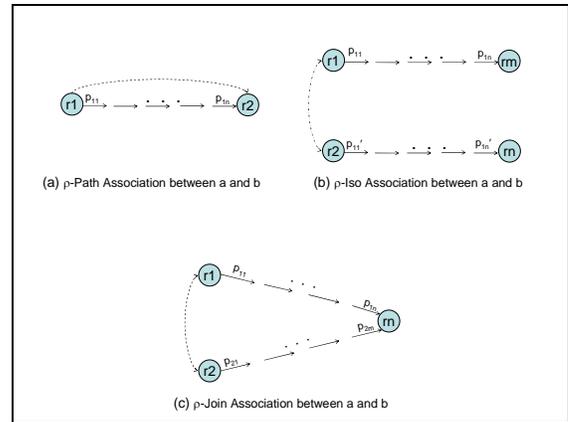

Figure 1: Semantic Associations

Figure 1(c) is called a `ρ-joinAssociation` between r1 and r2. The rationale for suggesting a relationship between r1 and r2 is that the paths $p_1 = p_{11}, p_{12}, .. p_{1n}$ originating from r1 and $p_2 = p_{21}, p_{22}, .. p_{2m}$, originating from r2 join at node rn. In other words, r1 and r2 meet at some point. Another type of association called a `ρ-isoAssociation` (shown in Figure 1(b)) indicates a similarity relationship between resources. The paths $p = p_{11}, p_{12} .... p_{1n}$ originating from r1 and $p' = p_{11}', p_{12}', ... p_{1n}'$ originating from r2 are semantically similar in that the corresponding edges in both paths are related in a subproperty relationship (i.e.,, for each p_{1i}, either p_{1i} is rdfs:subPropertyOf p_{1i}' or vice versa), therefore r1 and r2 are related by virtue of this similarity.

As mentioned earlier, Semantic Associations are based on the notion of Property Sequences. Let us assume the following interpretation functions for a class c and a property p, $[\![\ ]\!]^{\wedge}$, $[\![\ ]\!]^{+}$, and $[\![\ ]\!]$ such that:

i) $[\![\ c\ ]\!]^{\wedge}$ = { r | r is rdf:type of *c*} i.e., only proper instances of *c*

ii) $[\![\ p\ ]\!]^{\wedge}$ = {[r_1, r_2] | $r_1 \in \{\ \}\{[\![\ c\ ]\!]^{\wedge}$ | *c* ∈ *p*.domain }, $r_2 \in \{[\![\ c\ ]\!]^{\wedge}$ | *c* ∈ *p*.range } i.e., only proper instances of *p*

iii) $[[c]]^+ = \left\{[[c]]^\wedge\right\} \cup \left\{[[c']]^\wedge\right\}$ where c is rdfs:subClassOf c' i.e., proper instances of c and the superclasses of c.

iv) $[[p]]^+ = \left\{[[p]]^\wedge\right\} \cup \left\{[[p']]^\wedge\right\}$ where p is rdfs:subPropertyOf p' i.e., proper instances of p and the superclasses of p.

v) $[[c]] = \left\{[[c]]^\wedge\right\} \cup \left\{[[c']]^\wedge\right\}$ where c' is rdfs:subClassOf c i.e., all instances of c including its subclasses.

vi) $[[p]] = \left\{[[p]]^\wedge\right\} \cup \left\{[[p']]^\wedge\right\}$ where p' is a rdfs:subPropertyOf p} all instances of p including its subproperties,

A Property Sequence $PS = P_1, P_2, \ldots P_n$ is a finite sequence of properties whose interpretation is given by:

$$[[PS]] \subseteq \times_{i=1}^{n} [[P_i]] \text{ such that}$$

1. for ps an instance of PS, i.e., ps $\in [[PS]]$, ps[i] = [r_1, r_2] $\in [[P_i]]$ for $1 \leq i \leq n$. (We use the notation ps[i][0] and ps[i][1] to refer to r_1 and r_2 respectively.)
2. ps[i][1] = ps[i+1][0].

In the example above, Figure 1(a) shows a property sequence whose instance is the directed path from r1 to r2. . Figure 1(c) shows two property sequences p_{11}, p_{12}, .. p_{1n} and p_{21}, p_{22}, .. p_{2m} that form a ρ-joinAssociation between r1 and r2 and are called Joined Property Sequences while the two sequences of properties p_{11}, p_{12}, .. p_{1n} and p_{21}, p_{22}, .. p_{2n} in Figure 1(b) that form a ρ-isoAssociation between r1 and r2 are called ρ-Isomorphic Property Sequences.

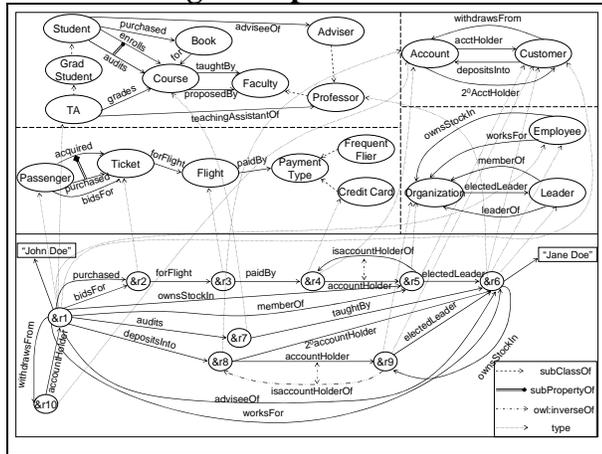## 2.1 Motivating Example

Figure 2: An example RDF knowledge base

We will motivate our work with a simple example. The top part of Figure 2 shows four schemas for four different domains University, Banking, Flight and Organization, while the bottom part shows a set of resources described using those schemas, i.e.,, the knowledge base. A Semantic Association search specifies a pair of resources and optionally some keywords, and the result is the set of ρ-pathAssociations, ρ-joinAssociations and ρ-isoAssociations that relate both resources. A semantic association search is sometimes referred to as a ρ-query because the search is done

using an operator ρ that extends traditional RDF query languages [3][2]. Table 1 below shows three Semantic Associations between resources r1 and r6, a student and her professor. The first row contains a ρ-pathAssociation that connects the student r1 to r6 via the purchase of a ticket that was paid for by r6 and row 2 connects r1 directly to r6 with an advisee relationship. Row 3 shows a ρ-isoAssociation in which both r1 and r6 have relationships that re similar, i.e.,, stock ownership. There are other semantic associations between r1 and r6 that are not shown in the table e.g. the ρ-joinAssociation involving the path in row 1 and the other paths that lead to r6.

Table 1: Example Semantic Associations

1	&r1 $\xrightarrow{purchased}$ &r2 $\xrightarrow{forFlight}$ &r3 $\xrightarrow{paidBy}$ &r4 $\xrightarrow{accountHolder}$ &r5 $\xrightarrow{leader}$ &r6
2	&r1 $\xrightarrow{adviseeOf}$ &r6
3	&r1 $\xrightarrow{ownsStockIn}$ &r5 $\xrightarrow{isAccountHolderOf}$ &r4 &r6 $\xrightarrow{ownsStockIn}$ &r9 $\xrightarrow{isAccountHolderOf}$ &r8

It is likely that the result of a search in a large knowledge base will be inundating, especially if the search includes multiple sources and destinations. This suggests a need to rank the results in some order of importance. On the other hand, it is not very clear how to ascribe importance to the results because of their complex and heterogeneous nature.

In the next section, we will discuss the relevance model called SemRank which addresses the problem of how to rank these results.

3 RANKING SEMANTIC ASSOCIATIONS

It would appear that there are different possibilities for ranking semantic associations results e.g., by shortest paths, longest paths, least frequently occurring paths, etc. However, each of these approaches makes an assumption about what is most relevant for *every* situation. In our experience, we have found that different applications have different needs and making assumptions that fix the ranking schemes can be limiting. Consequently, two main features of our ranking scheme are customizability (allowing users to select an appropriate ranking scheme) and flexibility (allowing users to easily apply different ranking schemes on results so that results may be viewed using different perspectives).

Fundamental to our ranking approach is the ability to measure how much information is conveyed by a result thereby giving a sense of how much information a user would gain by being informed about the existence of the result. This is closely related to the likelihood that a user could have guessed that such an association exists or the predictability of the association. Using such measure we may then rank results based on the search mode selected for the search. For example, if the context requires a conventional search then results that are deemed obscure and unpredictable will be ascribed the least importance. In Table 1, for example, we can say that the fact that a student is an advisee of a professor i.e.,, result 2, is probably the least surprising of the results and should be assigned the highest relevance when a conventional search is being performed. On

the other hand, when a discovery search is performed, the other two results are candidates for the most relevant result.

The factors used for measuring the predictability of a result include (i.) the uniqueness or `specificity` of the result and (ii.) how discrepant the structure of the result is from the possibilities that can be gleaned from the schema. In the case of specificity, it seems reasonable to conjecture that a commonly occurring association is more predictable than a rarely occurring association. The idea of uniqueness may be either with respect to the whole database or just to the set of resources of similar types. For example, it may be that the association is rare with respect to the entire database but frequent when considering only the set involving similar resources. We combine both notions of uniqueness to form a measure of information content of a result.

The issue of a result's discrepancy from schema arises because the multiple typing of resources by classes unrelated inheritance allowed by the RDF and OWL-Lite data models, often leads to paths at the data layer that cannot be predicted by just looking at the schemas. For example in Figure 2, the property sequence `purchased•forFlight•paidBy•accountholder•electedLeader` does not occur anywhere at the schema layer but occurs in a path from r1 to r6 because r4 and r5 are multiply classified. Deviations from schema represented paths are called `refractions` and paths with many refractions are unlikely to be easily anticipated by users, making them less predictable.

Finally, we allow users to optionally specify some keywords that capture relevance and results which contain semantic matches are ranked highest.

The rest of the section elaborates on these measures and how they are used to rank ρ-path associations. For brevity, we omit their application to other types of semantic associations.

3.1 Information Gain and ρ-Path Semantic Associations

In information theory, the amount of information contained in an event is measured by the negative logarithm of the probability of occurrence of the event. Thus if χ is a discrete random variable or an event that has possible outcome values x_1, x_2, …, x_n occurring with probabilities pr_1, pr_2, …, pr_n i.e., $Pr(\chi=x_i) = pr_i$, with $pr_i \geq 0$ and $\Sigma_{all\ i}\ pr_i = 1$, the amount of information gained or uncertainty removed by knowing that χ has the outcome x_i is given by

$$I(\chi = x_i) = -\log pr_i$$

The maximum information content of χ is attained when $pr_i = 1/n$ for all i. This is given by $I(\chi) = \log n$.

Based on this we can build a model for measuring the information content of a semantic association by considering the occurrence of edge as an event and RDF properties as its outcomes. We begin with defining the notion for a property and then extend it to a sequence of properties of a path. Assume that P is the set of all property types in a description base and χ is a discrete random variable with sample space $[[P]]$. Then for any valid property $p \in P$, the probability that $\chi = p$ is given by

$$Pr(\chi = p) = \frac{\left| [[p]]^{\wedge} \right|}{\left| [[P]]^{\wedge} \right|}.$$

We refer to this probability as the `specificity` SP of the property p. The specificity of a property is a measure of its uniqueness relative to all other properties in the description base. The information content of the occurrence of a property p $I(\chi=p)$ in the description base due to its specificity is:

$$I_S(p) = I(\chi=p) = -\log Pr(\chi = p)$$

It is possible develop a similar measure which exploits the semantics of RDF and RDFS. Given that RDF resources are typed, any two resources r1 and r2 have a finite number of valid properties from that may connect them. Using this information, we can estimate the information content of a property p linking r1 and r2 with respect to only the valid properties as possibilities. What we then expect is that information content will be larger in situations where the number of valid properties is large and smaller in situations with fewer valid properties.

The valid properties that can link two resources include those that have been explicitly defined in a schema and those that may be inferred from the semantics of RDFS. In particular, the semantics of multiple domains/ranges on a property p implies that a resource must belong to all the domain/range classes even if that membership is not explicitly stated. We introduce the concept of a `Representative Ontology Class` (ROC) as a concise summary of related classes by virtue of the equivalence of their interpretations. For example, in Figure 2 the classes `Book` and `Ticket` belong to an ROC which represents the set of things that can be `purchased`. We now make this notion more precise.

Definition 1 Representative Ontology Classes. For an ontology O with the set of classes C and properties P and $|C| = n$, let S be a $n \times n$ matrix with the following entries:

$$S_{ij} = \curlyvee p, c_i \in domain^{\wedge}(p) \wedge c_j \in range^{\wedge}(p)$$

where $domain^{\wedge}/range^{\wedge}$ refer to classes in the proper domain/range of a property (i.e., excluding their subclasses). We can define an equivalence relation ~ such that:

~(i, j) iff $S_{ik} = S_{jk}$ and $S_{ki} = S_{kj}$ for all k.

~ partitions the elements of S into the set $\mathcal{L}$ of equivalence classes of ~, where each equivalence class represents the set of classes that are equivalent with respect to their outgoing and incoming properties. It corresponds to the set of classes that should have the same interpretations. Each $l \in \mathcal{L}$ is called a `Representative Ontology Class` (ROC). The set of all possible properties `semLinks` that can directly connect two ROCs X and Y is given by `semLinks(X, Y)` $\rightarrow S_{i(X),i(Y)}$ where $i(l) \in \{i:C_i \in l\}$. $i(l)$ is called a `representative` for the ROC l and C_i is called a `member` of l. Since the members of each equivalence class are equivalent with respect to their outgoing and incoming properties, then it suffices to pick a representative class for X and Y, say $C_{i(X)}$ and $C_{i(Y)}$ respectively.

Now, given any two resources r1 and r2, we can measure the probability distribution of the types of properties that can connect them in the instance base. If we let π be the set of all possible properties that may connect r1 and r2, then π clearly depends on the types of r1 and r2. If C_1, C_2, .., C_m and D_1, …, D_n are the classes of r1 and r2 respectively and if X_1, X_2.. X_k, and Y_1, Y_2, ..Y_p are ROCs that C_i and D_j belong to respectively, then

$$\pi = \curlyvee semLinks(X_i, Y_j) \text{ and } \theta = \curlyvee \left\{ [[p]]^+ \mid p \in \pi \right\}.$$

θ represents the interpretation of all the valid properties that may connect r1 and r2. Thus, the probability that $\chi \in \theta$ is given by:

$$\Pr(\chi \in \theta) = \frac{|\theta|}{\left|[[P]]^{\wedge}\right|}$$

For a given valid property p in the description base, if $\chi \in \theta$, the probability that $\chi = p$ is given by:

$$\Pr(\chi = p \mid \chi \in \theta) = \frac{\Pr(\chi = p, \chi \in \theta)}{\Pr(\chi \in \theta)} = \frac{\left|[[p]]^{\wedge}\right|}{|\theta|}$$

We refer to this probability as the θ-Specificity SP$_\theta$ of property p. The θ-specificity of a property is a measure of its uniqueness relative to all other properties in the description base whose domain and range belong to the same ROC's, respectively. The information content of the occurrence of a valid property p $I(\chi=p \mid \chi \in \theta)$ in the description base due to its θ-Specificity can then be defined as

$$I_{\theta\text{-S}}(p) = I(\chi=p \mid \chi \in \theta) = -\log \Pr(\chi = p \mid \chi \in \theta).$$

To illustrate these concepts, for the property *"purchased"* connecting &r1 and &r2 in Figure 2 above, ROC$_1$ = {Student, Passenger}, ROC$_2$ = {Book, Ticket} with θ = {[[purchased]]$^+$, [[bidsFor]]$^+$}, so that the size of $\theta = |$ [[purchased]]$^{\wedge}|$ + | [[acquired]]$^{\wedge}|$ + | [[bidsFor]]$^{\wedge}|$. If there are 20, 40, 80 instances of the properties purchased, acquired and bidsFor respectively, with a total of 1000 property instances in the description base, then the specificity of purchased is 0.02 while its θ-specificity is 0.143.

We can extend these ideas to capture the information content of a ρ-path association. Let PS = p$_1$, p$_2$, …, p$_n$, be a property sequence and ps $\in$ [[PS]] a path. It is clear that ps occurs as frequently as the *least* frequently occurring property p$_i$ in PS. We define the information content of ps due to its specificity as $I_S(ps) = \max_{\forall i}\{I_S(p_i)\}$. Intuitively, this means that a path is as informative as its most informative edge with respect to the entire description base.

For the information content of ps due its θ-specificities we must somehow combine the different θ-specificities of the various properties on the path. However, since the θ-specificities of the edges are based on different probability distributions, we cannot meaningfully compared them with one another, so we must first normalize the values by taking the ratio of the observed information content to the maximum possible information content (as described earlier). This results in normalized θ-specificity values, NI$_{\theta\text{-S}}$. It is important that our combination function doesn't bias towards longer paths, therefore a sum function is not a good combination function. Also, we must ensure that the combination function distinguishes between paths with more uniform θ-specificity distributions than those with non-uniform θ-specificity distributions. To see why this is so, take for example two paths ps$_1$ and ps$_2$ that have the same average NI$_{\theta\text{-S}}$, but ps$_1$ has θ-specificity values for all its edges about equal while ps$_2$ has a range of θ-specificity values for its edges from low to high. Then it seems that ps$_2$ that has some edges with low information

content (i.e., some weak links) which should be easier to predict than ps$_1$ which has all its edges with an equal level of predictability. Therefore, to measure the information content of ps with respect to the θ-specificity of the properties p$_1$, p$_2$, …, p$_n$, we modify the value gotten from the average of the θ-specificities to:

$$I_{\theta\text{-S}}(ps) = \min_{\forall i}\{NI_{\theta\text{-S}}(p_i)\} + \frac{\left(\sum_{\forall i} NI_{\theta\text{-S}}(p_i)\right) - \min_{\forall i}\{NI_{\theta\text{-S}}(p_i)\}}{n-1}$$

This implies that information content due to θ-specificity is a combination of the information gained from the weakest point along the path and an average of the rest.

Finally, to the get the total information gained by knowing a path occurred we combine the values of information content due to both specificity and θ-specificity:

$$I(ps) = I_S(ps) + I_{\theta\text{-S}}(ps).$$

3.2 Refraction

As mentioned earlier, the multiple classification of resources allowed in the Semantic Web data models like RDF can create paths at the description layer the do not occur at the schema layer, especially when multiple schemas are used to describe a set of resources. We use the term Refraction to refer to a deviation from a path's representation at the schema layer. In other words, a description layer path starts and proceeds along exactly as described at the schema layer and then changes direction or refracts at some point.

In order to make this notion of refraction more precise, we need a representation of all the paths that are possible based on what is explicitly defined in or inferable from a schema. Then for a given path in the description base, any sequence of edges not represented would be considered a refraction due to a multiple classification of a node. We propose the notion of a Semantic Summary as such a representation. It is analogous to the concept of DataGuides [10] and other structural summaries [17] used to optimize the evaluation of path expression queries in semi-structured data models. A semantic summary is a graph in which the vertices are ROCs.

Definition 2 Semantic Summaries. A Semantic Summary for an ontology O with sets C/P of classes/properties is a graph $G_S = (V_S, E_S, \lambda, <)$

1. V_S is the set of ROCs of O as defined in definition 1.
2. $E_S = \left\{(x, y) \mid S_{i(X), i(Y)} \neq \varnothing \text{ and } x, y \in V_S\right\}$
3. $\lambda : E_S \to 2^P$
   i. $(x, y) \in E_S$, $\lambda(x, y) = semLinks(x, y)$
4. $<$ is a subsumption relation on nodes in V_S such that for two ROCs x and y, x is rdfs:subclassOf y if for c$_i \in$ x, $\exists c_j \in$ y such that c$_i$ rdfs:subClassOf c$_j$.

Two edges (u, v) and (w, x) of a semantic summary are said to be adjacent if v = w. Given a semantic summary $G_S = (V_S, E_S, \lambda, <)$ and a property sequence PS = p$_1$, p$_2$, …, p$_n$, if e$_i$, e$_j$ $\in E_S$ and e$_i$ and e$_j$ are not adjacent in G$_S$ and p$_i \in \lambda(e_i)$ and p$_{i+1} \in \lambda(e_j)$, then we say that there is a refraction from p$_i$ to p$_{i+1}$. Formally, for a path sequence PS = p$_1$, p$_2$, …, p$_n$, refraction(p$_i$, p$_{i+1}$) =

$$\begin{cases} 1 & \text{if } \exists e_i, e_j \text{ such that } e_i \text{ is adjacent to } e_j \ \wedge \ p_i \in \lambda(e_i) \ \wedge \ p_{i+1} \in \lambda(e_j) \\ 0 & \text{otherwise} \end{cases}$$

We use the term `refraction count` RC to refer to the number of refractions on a path, given by

$$RC(PS) = \frac{\sum_{i=1}^{n-1} \text{refraction}(p_i, p_{i+1})}{} \quad \text{for } n \geq 2, \ 0 \text{ otherwise.}$$

For example, in Figure 2, the path $p_{r1,r6}$ = $\&r1 \xrightarrow{\text{depositsInto}} \&r8 \xrightarrow{\text{accountHolder}} \&r9 \xrightarrow{\text{electedLeader}} \&r6$ with the property sequence PS = `depositsInto•accountholder•electedLeader`, refracts from accountholder to electedLeader because the resource &r9 is multiply classified as an instance of both `Organization` and `Customer` and RC(PS) = 1.

3.3 S-Match

In order to integrate IR style search with ρ-queries, we allow users to augment their queries with keywords. A Semantic Match (match of property or super/subproperty) of a keyword and a property occurring in Semantic Association) increases the rank value for that Semantic Association. The degree of the match and hence its S-Match value is determined by the proximity of the properties in the property hierarchy. This approach is very similar to that used in determining the similarity of concepts in an ontology [16] [21]. Given a property sequence PS = $p_1, p_2, p_3 \ldots p_n$ and a set of keywords K = {k_1, k_2, $k_3 \ldots k_m$}, the degree of a match between k_i and p_j is given by `SemMatch`(k_i, p_j) = $0 < (2^d)^{-1} \leq 1$, where d is the minimum distance between the properties in a property hierarchy. If two keywords match on the same property we take the maximum of their `SemMatch` values. Then for a path ps $\in$ [[PS]], its S-Match value is given by

$$\text{S-Match(ps)} = \sum_{i=1}^{n} \max_{j=1}^{k} \{\text{SemMatch}(p_i, k_j)\}$$

For example, in Figure 2 above, the minimum distance between *"audits"* and *"enrolls"* is 1, so that SemMatch(audits, enrolls) = ½. Given K = {audits, taughtBy} and a property sequence PS = `enrolls•taughtBy`, S-Match(ps) = ½ + 1 = 1½.

3.4 SemRank

All the factors discussed above when combined together give the SemRank value of a Semantic Association. However, the exact nature in which they are combined is dependent on the search mode μ which varies from 0 to 1, with 0 indicating purely Conventional and 1 indicating purely Discovery modes, respectively. Based on this, we build a `modulative` model for SemRank in which the mode μ specifies how each of the factors contribute to the rank of an association. The query mode μ modulates the contribution of the information content of an association to its rank as shown below:

$$I_\mu \text{ (ps)} = (1-\mu)(I(ps))^{-1} + \mu I(ps)$$

This leads to higher rank values being assigned to the most unpredictable paths at the purely discovery mode, and lower rank values being assigned at the purely conventional mode.

The query mode μ modulates the refractive count of an association as shown below RC_μ (ps) = μRC(ps).

Since predictable paths are desired at the purely conventional mode, paths ranked highest at this mode do not have refractions, as the formula above shows. Both the purely discovery and purely conventional modes seek to retrieve paths whose component properties have high S-Match values with the keywords provided by the user. Therefore, S-Match is not modulated by μ. The SemRank formula combines these three factors to assign a rank to any Semantic Association, adapting itself as the mode changes. It is defined for a ρ-path association ps as:

$$\text{SEMRANK(SA)} = I_\mu \text{ (ps)} \times (1+RC_\mu \text{ (ps)}) \times (1+\text{S-Match(ps)})$$

Using this model, we can provide a flexible ranking approach for ranking complex relationships.

4   ORDERING SEARCH RESULTS USING SEMRANK VALUES

The approach for obtaining an ordering on Semantic Associations resulting from a search will depend largely on the strategy for computing SemRank values. Possible strategies include integrating query processing, SemRank computation and result ordering into a single phase or performing the last two steps in a separate phase after query evaluation. The choice of the strategy to be adopted is dependent on whether exact orderings are required or whether approximately correct orderings are acceptable. In the case of approximately correct orderings, we trade correctness for efficiency. This happens because in the case of exact orderings it may be necessary to completely compute the SemRank values of all Semantic Associations and then sort them in order. When there is a large number in the result set, this may prove to be inefficient.

In this section, we will discuss an approach for computing SemRank values for Semantic Associations and an approximate Top-K ordering algorithm used in our SSARK system.

4.1 Overview of the SSARK System

The approach used in the SSARK prototype system implementation consists of three phases supported by the architecture shown in Figure 3.
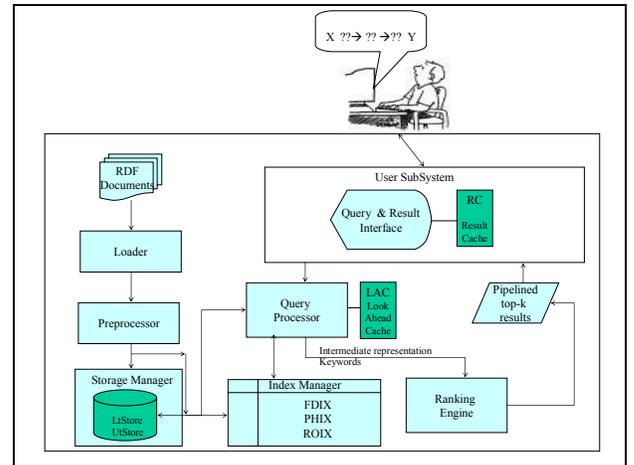

Figure 3: Architecture of the SSARK system

In the `preprocessing phase`, RDF documents are loaded and preprocessed into an intermediate representation by the `Loader` and `Preprocessor`. The intermediate representation of an RDF graph produced by the preprocessor is called a `path sequence`. A path sequence is a sequence of subgraph representations that is amenable to efficient query evaluation. The persistence of a path sequence is managed by the `Storage Manager` which allows for its storage in a database. When a query with a pair of resources is given, the `Query Processor` selects relevant subsequences of a path sequence and composes them to generate an annotated summary of the Semantic Associations called an `Annotated Path Expression Tree` (APET). The discussion of query evaluation is outside the scope of this paper. For the sake of brevity, the rest of the discussion will focus on ranking only ρ-pathAssociations, even though the other types of associations have also been investigated and are being prototyped.

The APET generated by the query processor is a tree representation of the regular expression that represents all paths found between the resources specified in the query. It is a `K-ary and-or` tree where leaves are the labeled edges of the paths and internal nodes are operator (union, concatenation) nodes with K children. Figure 4 shows an example APET. A semantic transformation process is performed during query evaluation to ensure that cycles are not represented in an APET. A discussion of the semantic transformation process is outside the scope of this paper but can be summarized thusly: For any cycle `c` with paths (a) from vertex v_1 to vertex v_2 and (b) from v_2 back to v_1, `c` is be broken up into two paths from v_1 to v_2. The first path is equivalent to (a) and the second is equivalent to $((b)^{-1})^R$ i.e.,, the reverse of the path (b) back to v_1 with the properties in (b) substituted with their inverse properties. Also during query evaluation the leaves of an APET (i.e., path edges) are annotated with their SemRank values. Then during the ranking phase, the `Ranking Engine` uses a pipelined Top-K algorithm to extract approximately the Top-K paths represented in the summary. The sequel elaborates on the structures used to support the SemRank computation as well as the Top-K algorithm.

4.2 Annotating Path Expression Trees
During query evaluation, the leaves of an APET are annotated with a set of values that contribute to their SemRank values. These values are either retrieved directly or computed from indexes in the `Index subsystem`. We will now summarize the roles of the indexes used in the computation of SemRank values:

- `Frequency Distribution IndeX (FDIX)`: FDIX maps each property p to a tuple $(|\ [[\ p\ ]]^{\wedge}\ |,\ |\ [[\ p\ ]]^{+}\ |)$ where $|\ [[\ p\ ]]^{\wedge}\ |$ is the size of p's proper extent and $|\ [[\ p\ ]]^{+}\ |$ includes the size of the proper extent of p's superproperties. These values are used for calculating Specificity and θ-Specificity.

- `Representative Ontology IndeX (ROIX)`: The Representative Ontology Index is a hierarchical index that maps resources to classes and then classes to ROCs. It also stores the semLinks that link the ROCs, i.e., the labels on the edges linking the ROCs in the semantic summaries. This information is used to determine the refraction count of a path.

- `Property Hierarchy IndeX (PHIX)`: Each property in the RDF data model may participate in a number of subsumption hierarchies. The idea is to index these properties in such a way that the distance between two entities in the hierarchy can be measured in constant time. To this effect, we index the properties in each hierarchy in a manner similar to the Dewey Decimal Coding (DDC) where each node is assigned an id that preserves its relative position amongst its siblings, prefixed by the id of its parent. For all the ids of all properties to be unique, each hierarchy is assigned a hierarchy id. Determining the distance between two properties then amounts to summing up the number of strings in the two ids beyond their Least Common Ancestor. For example given the ids 0.1.3.4.5.6 and 0.1.3.4.8, their LCA is 0.1.3.4. Beyond this, the first id has two strings (5.6) and the second has one (8), so the distance between them is three. Because a property may have more than one id, (since it may participate in more than one subsumption hierarchy), PHIX maps every property p to a set of ids. Using this index we can efficiently measure the distance between a keyword given a query and the properties on the resulting path which determines the SemMatch value of the path.

4.3 Retrieving Top-K results
After the query evaluation has returned an APET, the ranking engine extracts the top-K paths represented in the APET. An exact SemRank ordering may require an exponential time algorithm since all paths must be assigned a value first before the paths are ordered. Here, we use a practical approach that finds an `approximately correct` ordering thereby sacrificing total correctness for efficiency. The algorithm proceeds in two phases. In the first phase, the top-K paths are computed based on all values except the refraction count values. Then the second phase reranks the paths from the first phase based on their refraction count. It is clear that this will not always results in the totally correct SemRank ordering, but we expect that what we get is an approximation that is suitable for most applications. The reason for this is that during the top-K computation as will be discussed shortly, paths are composed iteratively into subpaths in a bottom up manner from the leaves so that the entire path is composed when the iteration is at the root of the APET. This means that properties of a path that are used in SemRank computation such as the refraction count can only be computed at the end of a path building phase as opposed to the other factors (e.g. specificity) that are properties of the edges themselves and are known once an edge is encountered.

The algorithm proceeds bottom-up computing top-k paths for nodes based on the top-K paths computed for its children. Each non-leaf node maintains a list for storing its Top-K paths, as well as a max-priority queue implemented using the heap data structure with which it orders its Top-K paths. The idea is to obtain the Top-K paths for each node by accessing only a minimal prefix of the Top-K paths of its children nodes, which are ordered in non-increasing order of SemRank values. At an 'or' node, during the first iteration of the algorithm, the first path from each child's Top-K paths are first enqueued into the max-priority q, then k paths are extracted from the queue. For any path extracted from the queue, the next path from its list is enqueued. For subsequent iterations, we update the queue with the first new entry of a list if it was updated after its previously

last entry had been enqueued. As such, not more than k paths are accessed from each child's list during each iteration of the algorithm. On the other hand, at an 'and' node, during the first iteration of the algorithm, the first path from each list is concatenated (preserving the order of the lists) to obtain the first of the k paths. To obtain the remaining k-1 paths, we first initialize the queue by obtaining and enqueueing a concatenation of the second path from each list with the first path from all other lists, then we extract k-1 paths from the queue. For each path p_i extracted from the queue, if p_i is a concatenation of paths $l_{a,1}, l_{b,2}, ..., l_{c,k}$ from lists $l_1, l_2, ... l_k$ respectively (where $l_{j,k}$ refers to the jth path from list k), we create a new path p_{i+1} by concatenating paths $l_{a_m,1}, l_{b_m,2}, ..., l_{c_m,k}$ (where $l_{j_m,k}$ is equal to $l_{j+1,k}$ if k equals m, and $l_{j,k}$ if k is not equal to m). Having created the path p_{i+1}, we only insert it into the queue if

- there does not exist any path $p_j = p_{i+1}$ with $l_{j_m,k}$ equal to $l_{j-1,k}$ when k equals m that is still in the queue and

- p_{i+1} is not already in the queue.

For other iterations, the queue may need to be re-initialized if it is empty. This algorithm is analogous to the ranked join algorithms described in [19] except that we have taken some measures to optimize queue operations. All possible paths have been extracted from the queue of both the 'and' and 'or' nodes when these queues become empty. In general, this technique can be applied to retrieve ranked paths from any tree representation of path expressions, irrespective of the relevance model used for ordering, as long as a monotone combining function is used in the join step. In the second phase, the rank of the Top-K paths retrieved from the first phase are re-computed this time including the refractive index of the paths, then the paths are re-ordered based on the new rank values.

To illustrate this, suppose we want to retrieve the top-2 paths from the APET shown in Figure 4(a) using the approximate retrieval technique.
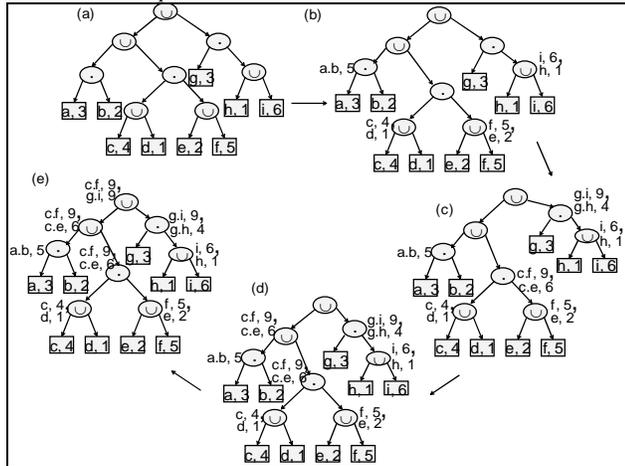
Figure 4: APET showing Top-K evaluation

For this example, we obtain the rank of a path by summing the ranks of its sub-paths. Figure 4(b) shows the state of the APET after all sub-trees of height 1 have been processed. Processing continues in a similar manner until the list of the root node of the APET gets updated with the Top-K paths. Figure 4(e) shows the state of the APET after the first ranking phase. The top-2 paths c.f and g.i both have the same ranks (9). If we assume that there is refraction from g to i and none from c.f, then during the second ranking phase, the ranks of these paths would be re-computed to incorporate refraction. The new rank of g.i would then be 18 (9×2), so that after they are re-ordered, g.i precedes c.f.

Given an APET, let R = $þ_1$, $þ_2$, …, $þ_n$ be all paths in the APET in non-increasing order of SemRank and let R′ $= þ_{1'}$, $þ_{2'}$, …, $þ_{k'}$ be the Top-K paths obtained from the APET using the approximate retrieval technique. We define the approximation error or cost of approximation as the average distance between the index of a path in R′ and its occurrence in R. This is given by

$$\frac{1}{k} \sum_{þ_{j'} = þ_i} | j' - i | .$$

5   EMPIRICAL EVALUATION

There are an increasing number of publicly available RDF data sets ranging from those that are narrowly focused (e.g. DBLP [30], ODP [31]) to those with broader scope covering multiple domains (e.g. TAP [29], SWETO [28]). However, most of these presented limitations that made them unsuitable as evaluation testbeds for SemRank. Because the SemRank is property-centric it was important that the testbeds have a wide variety of relationships. This was not the case in the narrowly focused ontologies, where most relationships tend to be of the same kind. In such a situation, there are little or no distinguishing features between the relationships. This was also present to some extent in the broader ontologies because they tended to be fragmented into smaller focused ontologies with limited connections between them. Also most of the ontologies were organized as hierarchies so that most of the relationships represented were inheritance relationships. Another important issue is that evaluation testbed(s) must have data distributions that model reality. This is because if a testbed's data distributions are skewed merely because the data collection process was not comprehensive then a SemRank value for a result may be meaningless (e.g. a property may be treated incorrectly as being rare even though the low frequency was due to the incompleteness of data). These problems are merely a reflection of the early developmental stages of testbeds for the Semantic Web. We expect that some of these ontologies will evolve into rich collections that may be very useful for future evaluations.

Consequently, the evaluation of SemRank discussed here was done on synthetically generated data. The data generation was guided by rules to ensure that data distributions mirror the real world. For example, in generating data relating to Students and Courses, it is often the case that the total number of students merely auditing the class, are less than 10% of the enrolled students. Our sample data builds on the schemas used in the

example in Figure 2 involving the University, Banking, Flight and Organization domains. An example query for the Semantic Associations between the resources r1 (Sarah White) and r6 (Zachary Black) is shown in Figure 5. It shows the simple query interface (sliding bar) that can be used easily to adjust search modes without users having to manipulate different criteria values. The same results are shown in using different search modes Figure 6 (purely conventional $\mu = 0$), Figure 8 (purely discovery $\mu = 1.0$) and Figure 7 (in between i.e., $\mu = 0.5$). A close examination of the results provide a justification for a modulative relevance model.
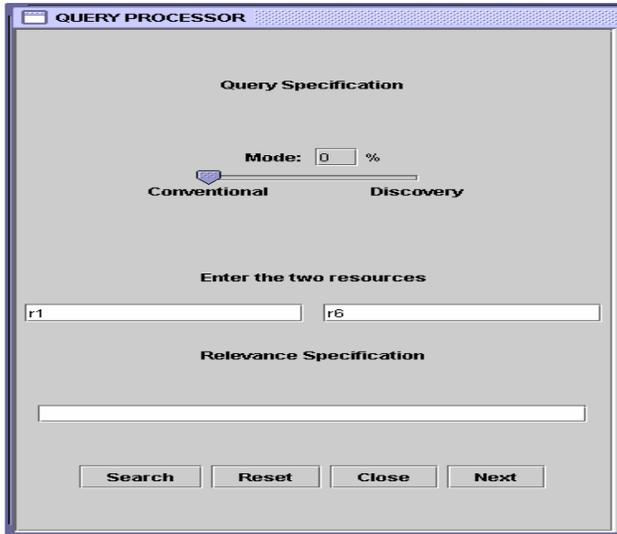
Figure 5: Query Interface

In addition, we can see that different orderings map closely to our intuition. The conventional search mode shown in Figure 6 returns as the first two results the paths of length = 1, both of which have the same number of possible valid properties in the schema (2 in this case. adviseeOf and TAOf for the first result, worksFor and ownsStockIn for the second).

Figure 6:Results of Search at $\mu = 0$ (Conventional Mode)

These results (advisee-adviser relationship between a student and a professor than an employee-employer relationship) map to our intuition as the most predictable relationships.

As earlier mentioned, SemRank orderings are independent of path lengths. For example the longest path which comprises both very common edges and very rare ones, with more of the former than the latter, is ranked third. This path which lies in between very rare and very common, is ranked first at $\mu = 0.5$, as shown in Figure 7 below.

Figure 7: Results of Search at $\mu = 0.5$

Ordinarily, one might expect that the orderings of results at the purely conventional ($\mu = 0$) and purely discovery ($\mu = 1$) modes would be exact inverses of each other. However, this is not always the case since refraction only plays a role at the discovery end of the search spectrum.

Figure 8: Results of Search at $\mu = 1$ (Discovery Mode)

For example, the ordering of path 1 and 2 at the purely discovery mode (ninth and eighth, respectively) is the exact inverse of their ordering at the purely conventional mode (first and second, respectively) because there is no refraction along both these paths. However the orderings of paths 7 and 4 (ranked third and fourth respectively at the purely conventional mode and fourth and third respectively at the purely discovery mode) conform to this intuition locally but not globally, since refraction occurs along both paths.

As mentioned earlier, users are allowed to augment their queries with keywords. To illustrate the effect any the use of keywords may have, Figure 9 below shows the same query at the purely conventional mode but now with the keywords {`enrolls`, `depositsInto`} supplied.

Figure 9: Results of Search at μ = 0 with keywords "enrolls" and "depositsInto"

Note that the keyword `depositsInto` has a higher S-Match value than `enrolls` since `audits`, a sub-property of `enrolls` appears along a path in the result, instead of `enrolls`.

With the S-Match value for these keywords, paths 4, 3, 9 and 5 ranked fourth, fifth, sixth and eighth respectively in Figure 6 are ranked second, fourth, fifth and seventh respectively in Figure 9.

These examples show that as long as the distribution of data reflects real world situations, the ordering of the results should be fairly close to user expectations.

6   RELATED WORK

[11] presents the idea of semantic search aimed at improving searches of documents on the Semantic Web by augmenting the results of traditional searches with relevant data obtained from multiple sources on the Semantic Web. [9] seeks to find important related nodes to a given set of keywords using a spread activation mechanism that is guided by information or knowledge provided by a domain expert/knowledge engineer. As we have discussed, discovering and ranking relationships

presents other challenges. In [26], the authors introduced the idea of ranking resources on the Semantic Web based on a set of semantic links that are used to describe the semantic relationships amongst the resources. Our idea of using the similarity of keywords provided by the user to enhance the importance of a Semantic Association is in a way similar to this. [25] presents an interesting ontology-based ranking scheme for ranking entities on the Semantic Web. This scheme determines the relevance of the entities based on their specificity. In our scheme we have also used the notion of specificity as a measure of the relevance of Semantic Associations. However our measurement of specificity of an association is different from that used in this work. Furthermore, our work differs from the above two in two significant ways. First, we focus on ranking Semantic Associations that exist between entities and not the entities themselves. Second, we focus on providing a flexible approach as opposed to a fixed approach to ranking these associations. In this way, the associations can be ranked to meet the needs of the user. To the best of our knowledge, the issue of ranking Semantic Associations on the Semantic Web has only been addressed by our colleagues in [1]. This work describes several criteria upon which relevance of Semantic Associations between entities are determined. Although the approach taken is flexible, it involves a user having to specify parameters for each of the criteria which can be an overwhelming task. Furthermore, our work significantly advances the understanding of the nature of the results.

7   CONCLUSION AND FUTURE WORK

We have presented an approach and framework for ranking complex relationships resulting from a "relationship search". Although an empirical evaluation was done using synthetically generated data due to the limitations of existing RDF data collections, it sufficed to show the justification for a flexible ranking approach that provides a variety of result orderings that a user may choose from, as opposed to locking users into a particular ranking scheme that may be unsuitable for the needs. In the next phase of our work, we will focus on performing evaluations on real world data.

8   ACKNOWLEDGMENTS

9   REFERENCES
[1] Aleman-Meza, B., Halaschek, C., Arpinar, I., and Sheth, A. Context-Aware Semantic Association Ranking. In Proceedings of SWDB'03: 33-50, Berlin, Germany, 2003

[2] Anyanwu, K., Sheth, A. The ρ operator: Discovering and Ranking Semantic Associations on the Semantic Web, ACM SIGMOD Record, v.31 n.4, December 2002

[3] Anyanwu, K., Sheth, A. ρ-Queries: enabling querying for Semantic Associations on the Semantic Web. WWW 2003. pages 690 – 699.

[4] Barton, S. Designing Indexing Structure for Discovering Relationships in RDF Graphs. DATESO 2004: 7-17.

[5] Booch, G. Object Oriented Design with Applications, Benjamin-Cummings Publishing Co., Inc. Redwood City, CA USA. 1990.

[6] Brickley, D., Guha, R.V. RDF Vocabulary Description Language 1.0: RDF Schema. W3C Working Draft, 2002.

[7] Brin, S., Page, L. The anatomy of a large-scale hypertextual Web search engine. WWW1998 pages 107--117. Brisbane, Australia.

[8] Cohen. S., Mamou, J., Kanza, Y., Sagiv, Y. XSEarch: A Semantic Search Engine for XML, VLDB 2003.

[9] Rocha, C., Schwabe, D., Poggi de Aragao, M. A Hybrid Approach for Searching in the Semantic Web. WWW2004.

[10] Goldman, R., Widom, J. Dataguides: Enabling query formulation and optimization in semistructured databases, VLDB, 1997.

[11] Guha, R. V., McCool, R., Eric Miller: Semantic search. WWW 2003: 700-709.

[12] Halaschek, C., Aleman-Meza, B., Arpinar, B., Sheth, A. Discovering and Ranking Semantic Associations over a Large RDF Metabase. VLDB 2004 demo paper.

[13] Hristidis, V., Papakonstantinou, Y., Balmin, A. Keyword Proximity Search on XML Graphs. IEEE ICDE, 2003.

[14] Kleinberg, J. Authorative sources in a hyperlinked environment. J. ACM, 48:604-632, 1999.

[15] Guo, L., Shao, F., Botev, C., Shanmugasundaram, J. XRANK: Ranked keyword search over XML documents. In ACM SIGMOD 2003, Pages: 16–27, San Diego, California.

[16] Lin, D. An Information-Theoretic Definition of Similarity, Proceedings of the Fifteenth International Conference on Machine Learning, p.296-304, 1998.

[17] Milo, T., Suciu, D. "Index structures for Path Expressions ". In Proc. of the 7th Intl. Conf. on Database Theory, January 1999.

[18] Mukherjea, S., Bamba, B: BioPatentMiner: An Information Retrieval System for BioMedical Patents. VLDB 2004: 1066-1077.

[19] Natsev, A., Chang, Y. C., Smith, J. R., Li, C. S., Vitter, J. S. Supporting incremental join queries on ranked inputs. In Proc. of VLDB 2001.

[20] Stojanovic, N., Mädche, A., Staab, S., Studer, R., Sure, Y. SEAL -- A Framework for Developing SEmantic PortALs. In: K-CAP 2001 – In Proc. of ACM Conference on Knowledge Capture, October 21-23, 2001.

[21] Rodriguez, M. and Egenhofer, M. Determining Semantic Similarity Among Entity Classes from Different Ontologies, IEEE TKDE , 15 (2): 442-456, 2003.

[22] Shannon, C.E. (1948), A Mathematical Theory of Communication, Bell Syst. Tech. J., 27, 379-423, 623-656.

[23] Sheth, A., Aleman-Meza, B., Arpinar, I. B., Halaschek, C., Ramakrishnan, C., Bertram, C., Warke, Y., Avant, D., Arpinar, F. S., Anyanwu, K., Kochut, K. Semantic Association Identification and Knowledge Discovery for National Security Applications. Journal of Database Management, 16 (1), Jan-Mar 2005, pp. 33-53.

[24] Sheth, A., Arpinar, B., Kayshap, V. Relationships at the heart of Semantic Web: Modeling, Discovering and Exploiting Complex Relationships. Enhancing the Power of Internet Studies in Fuzziness and Soft Computing. M. Nikravesh, B. Azvin, R. Yager and L. Zadeh, Springer-Verlag, 2003.

[25] Stojanovic, N., Studer, R., Stojanovic, L. An Approach for the Ranking of Query Results in the Semantic Web. ISWC 2003, Pages 500 – 516.

[26] Zhuge, H., Zheng, P. Ranking Semantic-linked Network, WWW2003, Budapest, May, 2003.

[27] Customer Identification and Risk Assessment (CIRAS), Semagix Inc.http://www.semagix.com/solutions_ciras.html

[28] SWETO: Semantic Web Technology Evaluation Ontology: http:/lsdis.cs.uga.edu/projects/SemDis/Sweto.

[29] TAP: http://tap.stanford.edu/.

[30] DBLP: An RDF ontology for DBLP. http://www.semanticweb.org/library/.

[31] ODP RDF dump http://rdf.dmoz.org/.