A Hybrid Approach for Searching in the Semantic Web

Cristiano Rocha
Dept. of Informatics, PUC-Rio and Milestone - I.T.
Rua Marquês de São Vicente 225
Prédio Gênesis, Sala 21b
Rio de Janeiro, RJ 22453-900, Brasil
+55 21 3114-1800
crocha@milestone-ti.com.br

Daniel Schwabe
Dept. of Informatics, PUC-Rio
Rua Marquês de São Vicente 225
Rio de Janeiro, RJ 22453-900, Brasil
55 21 3114-1500 ext 4356
dschwabe@inf.puc-rio.br

Marcus Poggi de Aragão
Dept. of Informatics, PUC-Rio
Rua Marquês de São Vicente 225
Rio de Janeiro, RJ 22453-900, Brasil
+55 21 3114-1500 ext 4339
poggi@inf.puc-rio.br

ABSTRACT
This paper presents a search architecture that combines classical search techniques with spread activation techniques applied to a semantic model of a given domain. Given an ontology, weights are assigned to links based on certain properties of the ontology, so that they measure the strength of the relation. Spread activation techniques are used to find related concepts in the ontology given an initial set of concepts and corresponding initial activation values. These initial values are obtained from the results of classical search applied to the data associated with the concepts in the ontology. Two test cases were implemented, with very positive results. It was also observed that the proposed hybrid spread activation, combining the symbolic and the sub-symbolic approaches, achieved better results when compared to each of the approaches alone.

Categories and Subject Descriptors
H.3.3 [**Information Systems**]: Information Search and Retrieval

General Terms
Algorithms, Experimentation.

Keywords
Semantic Web, Semantic Search, Semantic Associations, Ontologies, Network Analysis, Spread Activation Algorithms.

1. INTRODUCTION
With the currently growing interest in the Semantic Web, it is reasonable to expect that increasingly more metadata describing domain information about resources on the Web will become available. The idea presented here is to enrich the search process for hypermedia applications with information extracted from the semantic model of the application domain. One of the novelties in the semantic search proposed is the combination of spread activation techniques with traditional search engines techniques to obtain its results.

One of the greatest problems of traditional search engines is that they typically are based in keyword processing. Consider the following motivating example for a research institution domain, shown partially in Figure 1. This domain deals with people, publications and research areas. Notice that "Keyword" is not a concept of the model, but is used in the diagram to represent the fact that a keyword occurs inside the textual representation of the associated concept instances. For instance, the keyword "web" occurs inside the concept instance "The Evolution of Web Services" since it appears in the publication's "title" property. The keyword "ontology" is also related to the same concept since it appears in its "abstract" property.

A query with the keyword "web" would have as results only nodes of type Publication where this word occurs. If the user searches for nodes of type "Professor", the result could well be an empty set, since the keyword "web" may not appear inside the description text (page) of any of the professors. On the other hand, analyzing the semantics of this domain, it seems intuitive to think that if a given professor has many publications that are related to a given keyword, there is a great possibility that the professor himself is indeed related to that same keyword, and should be returned as a result of the query. Therefore, in the previous example, the Professor node "Schwabe" could be returned as a result for the query "web".

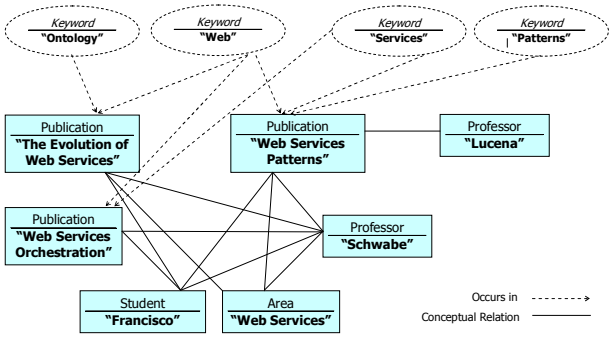

Figure 1. Semantic model instance

The semantic search proposed here is especially useful in applications where the user searches for concept instances of the model, instead of searching for "arbitrary" web pages. That is, usually the keywords in the query denote one or more concepts. These kinds of queries are described as research searches [9]. In other words, each page in the application is a hypermedia representation of an instance of a node in the model. More generally speaking, it is also useful when there is rich metadata associated with web pages.

Some existing semantic searches require the user to express its query in terms of other concept instances of the applications, imposing a high cognitive demand on the user. For example,

WWW 2004, May 17-22, 2004, New York, New York, USA.
ACM 1-58113-844-X/04/0005.

consider the model presented in Figure 1. In most semantic searches, if the user intended to search for students which are closely related to professor "Schwabe", he would have to write a query consisting of keywords (e.g., "Schwabe") but he would also have to tell the system that "Schwabe" is a concept of type "Professor". Having to know in advance the existing types in the domain of the application makes it more difficult for users to express their information needs.

The majority of users are accustomed to expressing their information needs in terms of keywords. It would be interesting to have a semantic search that has a traditional "Google-like" interface (keyword queries), but at the same time performs semantic processing. A semantic search that enables both textual information and RDF annotations querying is presented in [5]. Froogle [7] also presents a very interesting approach for product searches. It is a search engine specialized in querying for products, where the user expresses the products he wants to search for using keywords that are associated with the product (i.e. its brand, name, model, etc.). Froogle tries to guess the product the user wants to search for by associating the keywords in the query with the metadata that describe the products in their knowledge base. Another interesting semantic searcher is SCORE [15]. It uses automatic classification and information-extraction techniques together with metadata and ontology information to enable contextual multi-domain searches that try to understand the exact user information need expressed in a keyword query.

With these aspects in mind, we propose a semantic search that enables the user to express his information needs in terms of keywords, but at the same time uses the semantic information regarding the domain of the application to obtain results that are not possible in traditional searches. In traditional searches, a document is usually retrieved when at least one of the keywords in the query string occurs within it. The approach here is to obtain all concept instances that are related to a given word even if that word does not appear inside the concept. The system can infer relations through a spread activation algorithm, making it possible to retrieve concepts which do not contain any of the specified words.

As will be seen in the next sections, the proposed search showed to be particularly good in applications where the concepts in the ontology have rich and extensive textual information. For example, the Publication concept presented in Figure 1 has an "abstract" attribute which carries a lot of useful information.

The paper is structured as follows: in the second section we present some background on spread activation. In Section 3 we present the algorithm we developed. In Section 4 we propose a semantic search, whereas in Section 5 we present the results obtained in two existing applications. In Section 6 we discuss the architecture of the system and present some further thoughts on the results obtained. Related work is presented in Section 7 and Section 8 contains some concluding remarks and future work.

2. BACKGROUND

Spread Activation techniques are one of the most used processing frameworks for semantic networks. It has been successfully used in several fields, particularly in Information Retrieval applications [3][4]. Since it was developed in the Artificial Intelligence area as a processing framework for semantic networks and ontologies, we envision it to be a natural and interesting choice of knowledge processing algorithm in the context of the Semantic Web. An interesting and recent system that uses spread activation to process ontologies is ONTOCOPI [11], which tries to identify communities of practice within an organization.

The spread activation algorithm works basically as a concept explorer. Given an initial set of activated concepts and some restrictions, activation flows through the network reaching other concepts which are closely related to the initial concepts. It is very powerful to perform proximity searches, where given an initial set of concepts, the algorithm returns other concepts which are strongly connected to them. An overview of spread activation techniques is presented in [4].

Usually spread activation techniques are used either on semantic networks (where each edge in the network has only a label associated to it) or on associative networks (where each edge has only a numeric weight associated to it). In the Semantic Web, we use ontologies as the semantic network used by the spread activation algorithms.

Ontologies and their instances carry much more information than what is explicitly stated, as there is much "hidden" information entailed by the relations (i.e., a semantically-based link structure). In traditional ontologies, it is only possible to indicate the absence or presence of a relation between two concept instances; in many situations, it would be desirable to express some strength measure associated with the relation. The classical way is to associate a numerical value to the corresponding link.

One of the ideas in this work is to extract knowledge from the ontology and its instances in order to obtain a numerical weight for each existing relation instance in the model. The result is a hybrid instances network, where each relation instance has both a semantic label and a numerical weight. The intuition behind this idea is that better results in the search process can be achieved using the semantic information together with sub-symbolic (numerically encoded) information extracted from the instances. A similar idea was presented in [17], to provide a novel approach for ranking the results of ontology-based searching in the Semantic Web, with good results. We call Weight Mapping the technique of calculating a numerical weight value for each relation instance, based on the analysis of the link structure of the knowledge base.

In the next sections we will use the word node to denote a concept instance in the network; the word edge to denote a relation instance; and the word graph to denote the hybrid network.

3. ALGORITHM

We propose a hybrid spread activation approach consisting of the combination of the weight mapping techniques proposed here with traditional spread activation techniques.

3.1 Weight Mapping

As discussed in Section 2, it is necessary to associate a numerical value to each relation instance in the network. Different ideas were tested in devising a calculation that can generate a formula for the strength of each existing relation instance in the knowledge base. It is not possible to devise a formula that proves to be the best for all application domains. For example, in [17], a calculation for the relevance of a relation is presented. It is proportional to the specificity of all the terms in the relation.

We propose three different measures - cluster, specificity and combined - which we found very useful in developing our system. We are aware that the choice of these measures is totally

application and task dependent. In the next subsections we will use the word concepts as an abbreviation for concept instances.

3.1.1 Cluster measure

The first measure tries to establish the degree of similarity between two related concept instances in a relation. The similarity measure used is very similar to the cluster function used in [2], obtained by specializing that function for concepts that relate to each other. The formula below indicates the similarity between concept instance C_j and concept instance C_k.

$$W(C_j, C_k) = \frac{\sum_{i=1}^{n} n_{ijk}}{\sum_{i=1}^{n} n_{ij}}$$

The value n_{ij} represents the fact that concept C_j is related to concept C_i (it is 1 if the concepts are related and 0 otherwise). The value n_{ijk} represents the fact that both concepts C_j and C_k are related to the concept C_i (it is 1 if both concepts C_j and C_k are related to C_i and 0 otherwise). Therefore, the weight $W(C_j, C_k)$ represents the percentage of concepts that C_k is related to, given that C_j is also related. This measure is also similar to the confidence measure proposed in [16], which is widely adopted in algorithms for association rule discovery. The idea behind this measure is that concepts that share many common relations with other concepts are more similar. Another important point is that this similarity measure is asymmetric; the limitations in using symmetric similarity coefficients have been shown in [12].

3.1.2 Specificity Measure

The second measure is similar to the *idf* (inverse domain frequency) measure [18] widely used in Information Retrieval (although in I.R. the log function is normally used). This measure is useful when the user wants to give the semantics of specificity or differentiation to the relation. The following formula was used for the specificity measure:

$$W(C_j, C_k) = \frac{1}{\sqrt{n_k}}$$

The value n_k is equal to the number of instances of the given relation type that have k as its destination node. Therefore, the weight of the relation is inversely proportional to the number of relations with the concept C_k. If few concepts which have the same type as C_j are related to the concept C_k, then $W(C_j, C_k)$ will be high. Otherwise, $W(C_j, C_k)$ will have a small value. The formula used is slightly different from the one usually used in I.R. but showed better results in our experiments. The square root function was used to smooth the measure.

3.1.3 Combined Measure

The third measure is the combined measure, obtained as the product of the two previous measures. Its calculation resembles the *tf-idf* strategy [18] that is commonly used in classic I.R. Both measures described before can be used separately depending on the desired semantics. However, in the general case the combined measure, which is a multiplication of the two measures described before, proved to be the best one in our applications. The first term tells how similar the two concepts are. The second term tells how specific the destination concept is. One of the lessons learned in the Information Retrieval area is that there are various similarity and specificity measures as well as various ways of combining them. Up to now it has not been possible to prove that any of the possible combinations outperforms all others in a large set of experiments [19]. Therefore, other similarity and specificity measures might be used in the future to achieve better results.

The calculation of the weight mapping value for a relation instance can be context sensitive. Depending on the context, some relation types can be more important than others. In the measures proposed here, all types of relations were considered to have the same relative weight. As a future work, we want to allow the user to assign different weights to the types of edges in the ontology schema in order for the weight mapping calculation to be context sensitive. This can be implemented by changing the formulas to a weighted average.

3.2 Hybrid Spread Activation

There are several works in the literature that present spread activation algorithms in semantic [3] or associative [2] networks. However, there are few works that use both approaches together.

The hybrid spread activation is the main part of the proposed system, occurring in the hybrid instances graph, where relations (links) have both a label that comes from the ontology definition, and a numerical weight, which comes from the weight mapping techniques. The spread activation algorithm works by exploring the concepts graph. Given an initial set of concepts, the algorithm obtains a set of closely related concepts by navigating through the linked concepts in the graph. Inferences occur naturally in this process, since the result set may contain nodes that are not directly linked to the initial set of nodes.

It is necessary to configure the algorithm for each given domain, in order for the exploration of the concepts graph to make sense. The spread activation algorithm is domain dependent. The relative importance of a path when compared to a different path can only be determined in a well defined context. As will be shown in the next sections, some restrictions can be applied to the propagation to achieve what is called a Constrained Spread Activation. Among the possible configurations parameters are the relative importance of each relation type, as well as the constraints to be considered in the propagation (e.g., maximum path size, maximum node fan-out, etc.).

The algorithm has as a starting point an initial set of instances from the ontology, which will be called nodes from now on. These nodes have an initial activation value. The main idea is that during the propagation, other nodes are activated and at the end of the propagation a set of nodes and their respective activations are obtained.

The activation value of the initial set of nodes is given as an input parameter to the algorithm. Therefore, it is possible to set different weights to the initial nodes of the activation depending on the node's importance to the task being solved. This weight will be the node's initial activation value. If no initial value is supplied to the initial set of nodes, the algorithm considers each of these nodes as having the same weight (set to 1.0). All nodes which are not in the initial set have their initial activations set to zero.

The initial nodes are placed in a priority queue which is in a non-increasing order with respect to the node's activation values. The node with the highest activation value is then taken out of the queue and processed. If the current node passes through all the restrictions, it propagates its activation to its neighbors. Considering the origin node as i and the destination node as j, the propagation to the neighbors occurs according to the following formula, where I denotes input and O output:

$$I_j(t+1) += O_i(t) * w_{ij} * f_{ij} * (1-\alpha)$$

The contribution of *i* is added to the current input value of node *j*. Thus, the algorithm rewards those nodes which are reached through different paths, by adding the contributions of all its neighbors. This contribution is obtained by multiplying the output value of node $i(O_i(t))$ by the weight of the edge w_{ij} and by the factors f_{ij} and $(1-\alpha)$.

The output of a node is given by the function $O_i(t)$. Different functions were tested (sigmoid, linear and threshold). An identity linear function was chosen (the output of a node is equal to its input) since it provided the best results in the initial tests we performed. The value w_{ij} corresponds to the numerical weight of the relation obtained by the weight mapping. The value f_{ij} corresponds to the relative weight associated to the symbolic type of the edge that connects node *i* to node *j*. Types of edges which are defined to be more important have higher weights associated to them. More details are explained in the next subsection.

The α value corresponds to the percentage of activation that is lost, every time an edge is processed, effectively functioning as an attenuation factor. That is, for every propagation through an edge a lost of activation is considered. This is an interesting feature since shorter paths can have preference over longer paths. An α value of zero implies in a lossless propagation All these parameters are application and task dependent.

The neighboring nodes which are activated and are not currently in the priority queue are added to it, and it is then reordered. The node that was just processed is placed in the results list, containing all the nodes that have been processed and are the result of the spread activation process. It is interesting to observe that even those nodes which are already in the results list might have their activation values modified later by nodes that are processed in the future, since the contributions from all neighbors are considered. Therefore, the order in which the nodes are processed is not necessarily the same as their order in the result set of the search.

This process repeats itself until a defined state is achieved (a defined output size for example), or there are no further nodes to be processed in the priority queue. At the end of the algorithm the result set contains the nodes which are the result of the spread activation process ordered by their activation values. The algorithm processes each edge only once O(|E|). The priority queue used has an insertion complexity of O (log(|V|). Therefore the total complexity of the spread activation algorithm proposed is O (|E| * log |V|), where E is the number of existing relation instances (edges in the graph) and V is the number of concept instances (nodes in the graph). Figure 2 presents a pseudo code of the algorithm.

3.2.1 Integrated Approach

The graph where the activation propagates is a hybrid one, as was previously described. Typically, classical algorithms for spread activations either use symbolic or sub-symbolic edges. In order to accomplish a hybrid propagation it was necessary to combine and integrate both types of information.

The integration strategy chosen and implemented was to map every symbolic edge into a sub-symbolic edge. A knowledge engineer is responsible for giving a numerical relative weight for each type of relation in the ontology. This is done based on the understanding of which are the preferred paths that should be used in order to solve the desired task. He also has to understand the relative importance of each relation type and has to propose weights that reflect this. This process may be difficult, and a fine tuning may be necessary through testing. In spite of this potential difficulty, for the applications tested in our work it was not necessary to use this relative weight to achieve good results – all relation types had the same relative importance (equal to 1.0).

```
List spreadActivation( VertexPriorityQueue input )
    List output; RestrictionsSet preRestrictions; RestrictionsSet posRestrictions;
    ActivationFunction activationFunc; double distanceDecayFactor;
    while( ( input.isNotEmpty() ) && ( stop != STOP_SPREAD_ACTIVATION ) )
        currVertex = input.removeMax();
        if( checkRestrictions( preRestrictions, currVertex ) == true )
                activation = activFunc.getActivation( currVertex );
                currVertex.visited = true;
                for( every edge e / Orig(e) == currVertex )
                        destVertex = e.getDestination();
                        edgeType = e.getType();
                        deltaInput = activation * e.getWeight() * edgeType.getWeight();
                        deltaInput *= ( 1 - distanceDecayFactor );
                        destVertex.input += deltaInput;
                        destVertex.activation = activFunc.getActivation( destVertex );
                        if( destVertex.visited == false )
                                input.addVertex( destVertex );
                outputQueue.insertVertex( currVertex );
        stop = checkRestrictions( posRestrictions );
    return outputQueue
```

Figure 2. Pseudo code of the algorithm

The weight proposed for each relation type is a real number greater than or equal to zero. The bigger the value, the more important the particular edge type is to the propagation. A value equal to zero implies that activation does not propagate through that edge type. This is very useful, since it makes it possible to prevent the exploration of paths that do not make sense in a given task. Therefore, every relation instance or edge in the graph has its weight defined by the given relative weight value associated to the relation type multiplied by the weight of the relation instance calculated by the weight mapping algorithm.

With this integration, the hybrid graph is transformed into an associative network where all the edges are sub-symbolic. At first glance, it might appear that from now on the propagation is not hybrid anymore, since the semantics of the relations were transformed in numerical information. This is not really true, since there are various constraints that can be configured in the spread activation which use symbolic information regarding the node instances, explained next.

3.2.2 Constraints

One of the problems of spread activation algorithms is that if the propagation is not well controlled, it might reach the entire network. To solve this kind of problem the activation is spread according to rules enforcing constraints. The constraints can be verified before (pre-constraint) or after (pos-constraint) a node is processed. Some of the possible constraints in our system are:

- Concept type constraint: this constraint is used when the activation must not propagate through nodes of a given concept type.
- Fan-out constraint: the spread activation should cease at nodes which are connected to more than a given number of other nodes.

- Distance constraint: the spread of activation should cease when it reaches nodes that are more than a given distance (threshold) from the initial set of nodes. In our experiments the maximum distance used was three.

4. SEMANTIC SEARCH

The general architecture of the proposed search is shown in Figure 3. The first two steps happen exactly in the same way as in traditional searches. The user expresses his query in terms of keywords that are fed to a traditional search engine. This search engine has access to all the existing nodes contents (i.e., the data associated with the metadata denoted by the node) in the knowledge base. To achieve this, for each node in the knowledge base, a node consisting of the concatenation of the values of all its properties is created in the instances graph. The traditional search happens on the contents of the nodes on the instances graph. Figure 4 shows the process of creating the instances graph from the existing instances. From the figure we can see that for each instance in the ontology its representation in the instances graph contains all its properties.

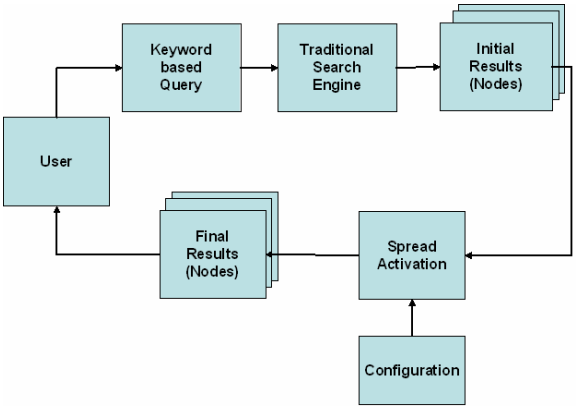

Figure 3. Semantic Search Architecture

The result given by the traditional search engine is a set of node instances ordered by their similarity with the query. This set of nodes is supplied to the spread activation algorithm as the initial set of nodes for the propagation. In addition, the ordering information given by the traditional search engine is also used. For each node, the traditional search engine provides a real number that measures the relative importance of that node with respect to the given query. This numeric value is used as the initial activation value for the node. Therefore, the nodes that were ranked well by the traditional search engine will have priority in the propagation since the exploration starts at the nodes with the highest activation values.

The spread activation occurs in conformance with the specified configuration using the domain semantics to traverse paths in the graph. The set of nodes obtained at the end of the propagation are presented to the user as the result of the semantic search. An interesting aspect is that the final list of results will not necessarily have any similarity with the initial set of results provided by the traditional search. Nodes that are not in the initial list of results might be in the final list of results and vice-versa.

It is important to observe that the transitivity might not always be valid in the model. That is, depending on the task to be accomplished by the user, it might be the case that some specific paths in the network should not be explored. It may also be the case that some paths are more important than others. This type of knowledge regarding the domain is defined in the spread activation configuration. This configuration should be done by a knowledge engineer specialized in the given domain and in the tasks that the system should help the user to accomplish. The knowledge engineer will set these configurations by defining the relative importance weights for each relation type as it was explained in Section 3.2.1. This configuration is done through a simple XML configuration file. In spite of this flexibility, our experiments show that it is possible to obtain good results with almost no specific tuning (see Section 6).

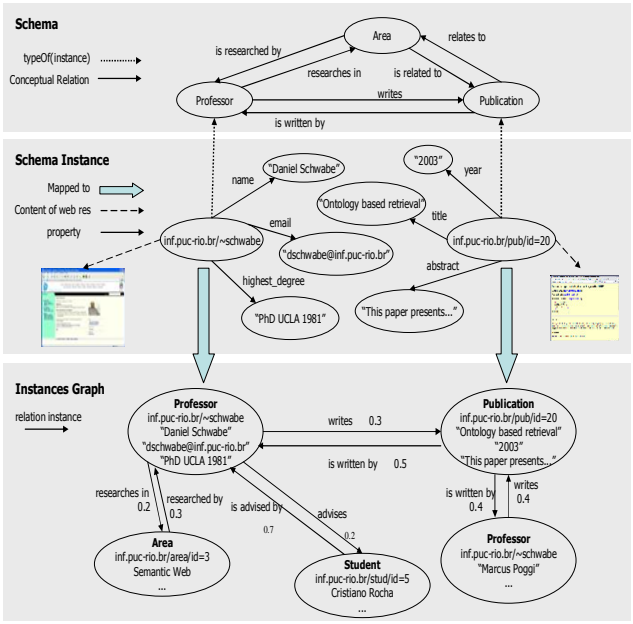

Figure 4. The Instances Graph creation process

Another important fact that should be mentioned is that the result of the semantic search is also ordered by the importance of the nodes for the given query. For example, considering the domain in Figure 1, if the query consisted of the words "ontology" and "web", the result set would probably bring the node "Professor Schwabe" ranked better than node "Professor Lucena", since the first has three publications related to the keywords and the latter has only one. This result is directly connected to the principle of activation sum that defines that a node which can be obtained through different paths should have all the contributions summed.

An interesting functionality is that the system provides, for each node obtained as a result of the propagation, the shortest path from one of the origin nodes. This allows recovering the path followed in the graph to obtain the inference. The node which contributed most for the activation of each node in the result set is also provided. Both these data are very important since they allow the knowledge engineer to better evaluate the results presented, and tune the search engine when necessary. If the results are not satisfactory, this information gives clues that show where the configuration should be changed in order to obtain better results.

The spread activation also enables the user to filter the searches. For example, the user might only be interested in results that belong to a given concept type (for instance, only nodes of type "Professor").

5. RESULTS

5.1 Department of Informatics Web Site

The first application used for testing was the web site of the Department of Informatics at PUC-Rio [14]. In this web site it is possible to obtain information about the main research areas, professors, projects, students, labs and publications. The knowledge base has around 2,630 node instances together with 6,554 relation instances. A small part of the research ontology used in the Web Site is shown in Figure 5.

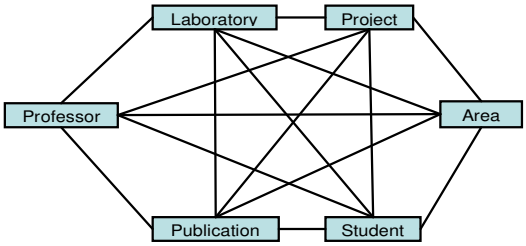

Figure 5. Research domain ontology

In the website of the department there are ten research areas, such as Software Engineering, Artificial Intelligence, Databases, etc. Due to the small number of areas, they have a broad meaning, and each one has more specific sub-areas. For example, the Software Engineering area has various sub-areas such as Requirements Engineering, Multi-Agent Systems, Frameworks, etc.

Imagine the situation of a potential student who wants to join the department in order to obtain his PhD. This student would like to find out which professor could be his advisor in the theme of Requirements Engineering. This area is a sub-area of Software Engineering, and is not listed as an official area in the "Areas" section of the department site. Consequently, the user has no way of directly obtaining the professors that have published more in this theme, or other students that are studying this theme, or even to find out which labs he could apply for, doing research in this specific field. To obtain this information, the student would have to do a traditional search for the keywords "Requirements engineering" inside the website, and navigate through all the resulting nodes. For example, he could analyze all the publications found, and try to find out who is the professor more active in this theme. This would be a hard task, since there might exist hundreds of publications in a given theme, and manually browsing through all of them would take an impractical amount of time.

To analyze the results generated by the semantic search in the context of the Department of Informatics at PUC-Rio (DI-PUC-Rio), two types of tests were made. The idea of the first type of test is to focus on searches where the words present in the query represent one theme in the computer science area. A dynamic page is generated for the given theme, linking it to all the other node types in the site. That is, having a theme as the input to the query, the system can provide as a result a page with the main professors, areas, publications, labs, students, projects and products related to the given theme.

It is important to notice that a professor that has many publications in the "Requirements Engineering" theme might be obtained as a result for a search for "Requirements Engineering", even if these words do not occur inside his homepage (or descriptive page). The spread activation algorithm is able to navigate the concepts graph and conclude that the professor is highly connected to that theme. It is not necessary for a keyword to appear inside the instance of a node in order for that node to be retrieved as a result.

The main objective of this first test is to analyze if the semantic search functionality works in the desired way. That is, beginning with a theme expressed in the query, it is possible to obtain the main nodes related to that theme inside the Informatics Department web site (students, professors, laboratories, publications, etc.). Since typically the words that describe a theme only appear inside the publications pages, it will be possible to analyze the inferences that make it possible to obtain other types of nodes as results. The goal is to analyze for this specific application if the user can obtain all the information regarding a theme directly through the search. To analyze the results we used expert users in the chosen themes. They were able to analyze if the functionality achieves the proposed objectives.

Twenty different themes were chosen to be analyzed (among them "lua", "oohdm", "corba", "frameworks", "aspects", "prefix codes", "Steiner problem" and "requirements"). Some themes needed more than one word to represent them. The themes were randomly chosen. We also tried to search for very specific themes that had no corresponding instance node representing them in the knowledge base. The availability of experts in the themes was also considered in the choice.

The analysis of results for semantic searches is still an open issue. There are no standard accepted measures such as recall and precision in the traditional Information Retrieval field. We searched the literature on semantic search for a reasonable measure, and did not find one that could be used in our case. Most of the published semantic search works present only their approach to the problem without analyzing the results obtained. We decided to do a qualitative analysis where, after analyzing the results, the expert was expected to say if he was satisfied with the results and if they seemed to help him in the proposed tasks.

The analysis was carried out as follows. Each of the suggested lists for a given theme (list of professors, list of publications, list of students, etc.) was separately analyzed by an expert in the given theme. This expert analyzed the contents of the list to check if the most relevant nodes were contained in the answer. The ordering of the result set was also analyzed. For all the chosen themes the results were considered good. All the generated lists for the test themes were analyzed positively by the experts generating an approval of 100%.

Figure 6 presents the result for one of the test cases used. The query consisted of the keyword "oohdm" which stands for Object Oriented Hypermedia Design Method. It is a method for designing and developing multimedia applications created at PUC-Rio and therefore actively researched in the Informatics Department.

When the user chooses to navigate to one of the nodes in the result set, the systems provides the shortest path followed by the spread activation algorithm to reach that given node. It also provides the node which contributed most to the activation in order for the current node to appear in the result set. Figure 7 presents an example. The student node "Gustavo Rossi" obtained as a result of the previous query (Figure 6) is being visualized by the user. In this case, the shortest path to it has size one and was obtained through the publication "*Systematic Hypermedia Application Design Using OOHDM*". Besides that, the professor

node “Daniel Schwabe” was the node which contributed most for the student’s final activation value since professor Schwabe is the most strongly related professor to the OOHDM methodology (in particular, he created it together with the given student). This information can be found on the upper central portion of the result page shown in Figure 7.

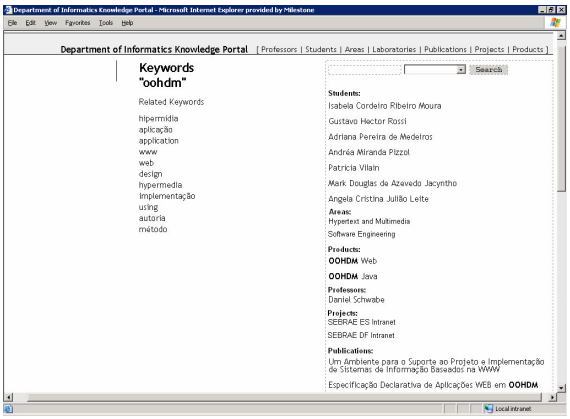

Figure 6. Search results for the query “oohdm”

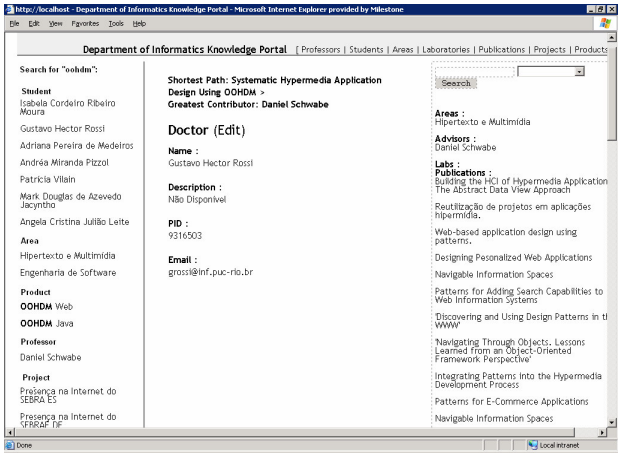

Figure 7. Visualization of one of the instances of “Student” returned by the search “oohdm”

The second type of test proposed uses more than one theme in the query. The idea is to make an initial query which is very general and contains many professors, students, labs, etc., in the result set. A successive goal is to analyze if, as another theme is added, those instances that relate to both themes obtain a better ranking when compared to the ones that only relate to the initial theme. This will make it possible to show that the combination of different propagation paths really works as desired, rewarding those nodes which are achieved through different paths, since all the activation values are added.

Therefore, after entering a set of themes in the query, the retrieved nodes should be those nodes which are strongly related to the majority of themes. It is interesting to note that a node which has a very strong relation with only one of the themes might eventually be better ranked than a node that has a weak relation with two or more themes.

The test cases consisted of queries which had up to four different themes. For each test case, the number of queries posed to the system was equal to the number of themes in the query. For example, in the test case (hypermedia, oohdm, framework, J2EE), four queries were made. The first one consisted only of the keyword “hypermedia”. The second query consisted of both “hypermedia” and “oohdm” keywords. This process of including new themes was repeated until all the themes of the test case were included, and the last query consisted of all the themes together. In each step of the process, the changes in the lists obtained were analyzed to check if the search worked as expected. The results obtained were also very good. At each addition of a new theme, the elements ranked first were the ones that were strongly connected to the set of themes.

Figure 8 shows the result for one of the test cases queries. In this case, the words “framework” and “J2EE” were added to the original query which consisted of the keyword “oohdm”. It is interesting to observe the change in the results when compared to the results in Figure 6 (query consisting of the keyword “oohdm”). The nodes most related to all three themes were ranked first. In particular, the “Student” instance “Mark” jumped from 6th to 1st in the student’s ranking. This happens because his thesis was about a J2EE framework for the design and implementation of applications based on the OOHDM method.

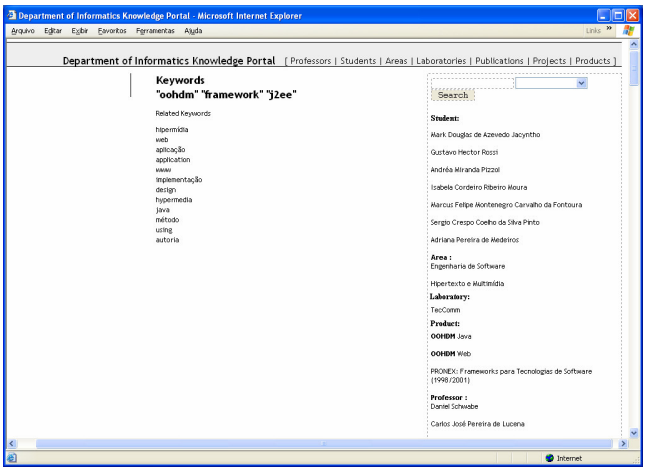

Figure 8. Results for the query “oohdm” “framework” “j2ee”

5.2 Portinari Web Site

The second application used for testing was the Portinari Project Web Site [13]. This site is an extensive knowledge base regarding the artwork, life and times of the most famous Brazilian painter, Candido Portinari. A part of the Portinari Project ontology is shown in Figure 9.

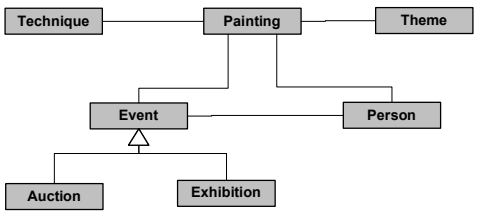

Figure 9. Portinari application ontology

The knowledge base has, among other items, information about all his paintings, as well as all exhibits where they were presented. It also has a list of all documents (books, articles, etc.) which mention any of his paintings, persons which are related to his work, and other types of information. The node “Painting” has an attribute which is the textual description of the painting image.

This textual representation is made by experts in the field and describes all the elements that exist in the painting (objects, colors, materials), using a controlled vocabulary. This description is extremely detailed and non-subjective. The knowledge base has around 16,000 node instances, together with 30,000 relation instances.

One of the most interesting aspects of this application, as far as testing our approach, is that this project has a group of experts that has a deep understanding of Portinari's work, and the existing information in the website. Therefore, it was possible to analyze and validate the results obtained with this group of experts.

The tests made were very similar to the ones made in the D.I. website. In the first type of test, 20 random keywords describing themes were used. Among these themes there were objects, animals, colors, Brazilian festivities, places, etc. It was possible to search for paintings, events, techniques and persons related to a particular keyword. For example, when querying for the word "carnival", it was possible for the user to retrieve not only the paintings in which Portinari portrayed carnival activities, but also the techniques the painter used most often when painting such pictures. The user could also retrieve the exhibits which had many paintings depicting carnival. This kind of chaining in the search process was highly appreciated by the experts in the project. They were especially impressed by the easy (keyword, Google-like) interface for retrieving this kind of information. The results were analyzed positively in 90% of the tests. In the few tests which the users were not satisfied, the main complaint was about the ranking of the retrieval and not about its contents.

The second type of test consisted of various themes (or keywords representing themes) in the same query. The results were also very good, and similar to the ones obtained in the first test.

6. DISCUSSION

The presented system was implemented in Java. We used the Lucene [10] search engine as the traditional search engine used in the first phase of the semantic searcher. The implemented system supports the full process of developing a semantic searcher for a new application. The process consists of the following tasks:

- Importing the Ontology and KB
- Weight Mapping Configuration
- Hybrid Spread Activation Configuration
- Integration with the Application

The system has an internal representation of the ontology and its instances. This makes it possible for the system to import ontologies defined in different languages by creating specific adapters that convert it to the internal representation. The two applications considered in the tests had their semantic model and instances defined in a language that can be easily mapped to RDF.

As previously mentioned the spread activation algorithm has many possible configurations and is domain dependent. Therefore it would be very interesting if the developed system could be easily extended, as needed, by other applications. We developed the system as a framework [6] given the need for great flexibility. Consequently, the addition of new constraints, filters, weight mapping measures, etc., is easily achieved since they were all designed as hot-spots of the system framework. The whole system consists of 75 classes. Figure 10 presents an overall picture of it.

In the Portinari application, the proposed search enabled the user to obtain information that even the domain experts were not able to provide. For example, it is now possible to enter a color name and search for the techniques that are strongly linked to the paintings where Portinari used the given color. It is also possible for a user to search for events where paintings depicting a given set of real world artifacts (e.g., table, hat, etc.) were exhibited. The best part is that the query is based on keywords and the user has only to type the names of the artifacts he wants. Since the ontology of the given domain does not define the concept of "artifact", this kind of search would not be possible using only ontology-based search. It would also be impossible to obtain this kind of result from classical search engines, except through a complicated manual chaining of queries.

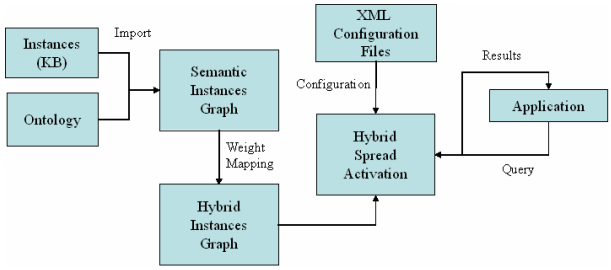

Figure 10. Implementation architecture of the proposed system

An interesting fact occurred in some test cases. Sometimes, the user of the application thought that an instance had been incorrectly retrieved. In the majority of cases, analyzing the given result (since the spread activation algorithm provides the path through which the node was obtained) the user convinced himself that the system was correct. This is one of the main virtues of the system. It can justify the reason why a certain instance was obtained as a result of the search.

Another important fact to notice is that in both applications studied the configuration used was a very standard one. Even though different weights for different paths in the graph can be configured, this feature was not used at all. Therefore, all the relations had the same importance weighting factor in the spread activation algorithm (all set to 1.0). It was not necessary for a knowledge engineer to fine tune the propagation in order to obtain very good results. This is a very important result since the success of the proposed system did not rely on the skills of the knowledge engineer in configuring the algorithm. We also believe that some profiles for a set of important and known domains could be stored with pre-defined settings fine-tuned to the given domain. Table 1 contains a summary of the evaluation results for the tests.

Table 1. Summary of the results obtained

Application	Test Type	Number Instances	Positive Evaluation
DI Website	1	20	100%
DI Website	2	10	100%
Portinari	1	20	90%
Portinari	2	10	100%

To compare the proposed hybrid approach with the traditional approaches, the same set of tests was carried out using the semantic network of the ontology without the weight mapping functionality. That is, the activation was propagated in a symbolic graph where the edges had only labels associated to them. The results obtained were poor. Most of the results obtained had no discernible relation with the given queried theme. To achieve better results with the symbolic propagation, it was necessary to

do a lot of tuning in the propagation, informing the algorithm which were the most important paths to be followed. It required much effort to achieve some good results in this scenario. In spite of achieving some good results, the results obtained through the hybrid approach were always superior, and required much less effort from a knowledge engineer as well.

The results obtained lead to the conclusion that the weight mapping technique somehow extracts some of the semantics of the domain and make it explicit through the weights that it generates for the relations. Consequently, the hybrid spread activation algorithm uses this information to achieve better results.

These results are indicators that the hybrid propagation is better than the semantic propagation. Besides that, the hybrid propagation is also better than the sub-symbolic propagation alone, since it makes it possible to configure preferable paths when needed. These arguments lead us to the conclusion that the hybrid spread activation performs better than the two other propagations alone. However, to really validate this conclusion, further tests with other applications and domains should be made.

The results shown here are based on the two applications described. We intend to implement this semantic search in applications with larger ontologies, to evaluate how scalable the proposed algorithm is. In both applications tested, the experienced response time for the proposed semantic search was slightly higher than the response time for a traditional search engine. We are aware that spread activation algorithms are highly costly and can sometimes be impractical for certain types of applications. Nevertheless, we envision it to be useful in various real-world applications as the ones presented in this paper.

7. RELATED WORK

An interesting approach for ranking query results using semantic information is presented in [17]. It considers other important sources for determining the relevance of results to a query such as the structure of the underlying domain and the characteristics of the search process. The content of the information repository is used in searching in a similar way to the weight mapping techniques proposed in this paper. They calculate the relevance of a relation instance for the user's query using a measure that is similar to the specificity measure proposed in this paper. However, the measure proposed in their work is symmetric. The ranking proposed in their work also takes into consideration the path used by each inferred result in order to calculate its relevance to the user query. This idea is also present in our work, since different weights can be assigned to the various paths in the propagation process. In their work though, the query is expressed through instances of the ontology and not through keywords.

A different semantic searcher is presented in [9]. It is built over the Semantic Web infra-structure and is designed to improve traditional web searching by augmenting traditional search results with relevant data aggregated from distributed sources. The query is also made through keywords that are mapped to existing concepts in the underlying ontology (through the TAP [8] search interface). The idea of navigating through the instances graph is also used in their work, although it is only used to augment the search results by presenting other concept instances strongly connected to the concept searched for (through a breadth-first search). That is, in their system a search for a singer might bring back not only the pages retrieved which mention that singer, but also some semantic data such as a list of the recordings by the singer, or his next tour dates. The idea of navigating through the graph to perform inferences is not pursued in their work.

Another interesting work related to our paper was presented in [5]. QuizRDF is a system which combines traditional keyword searches with the possibility of querying and navigating through the RDF annotations of the resources when they exist. The RDF information is indexed together with the textual information of the resources. The resulting index makes it possible for the system to search for keywords inside both the textual information of the resource and its RDF annotations. The possibility of mixing traditional information retrieval with semantic information retrieval is a common aspect with our work. However, the authors of the paper pointed out that one of the limitations of their system was that the user could not ask queries involving any "chaining" such as "Find me all instances of class painting painted by an (instance of class) painter whose first name is Pablo". This type of query is trivially mapped and answered in our system, because of the use of spread activation techniques.

ONTOCOPI [11] presents an approach similar to ours for processing ontology-based information through spread activation techniques. The proposed system is applied for identifying communities of practices (COPs) in an organization. ONTOCOPI attempts to uncover informal COP relations by spotting patterns in the formal relations represented in ontologies, traversing the ontology from instance to instance via selected relations. The activation in their system is propagated through a semantic network only, and there exists no idea of extracting semantics from the link structure like the weight mapping techniques proposed in our work. Their work also uses the spread activation system in a much narrower scope than the system proposed in this paper. We believe that our system could be successfully used for the same task as ONTOCOPI.

The work presented in [1] describes a framework where ranking techniques can be used to identify more interesting and more relevant Semantic Associations in RDF graphs. That is, associations among concepts can be calculated and given a ranking. This is done by analyzing the sequence of interconnected links in the path that relates two instances. It calculates a weight for a relation instance using a ranking strategy which is based on various aspects such as Subsumption Weight, Path Length Weight, Context Weight and Trust Weight. This ranking is calculated based on operations in the ontology model or schema graph, and not on the instances graph, as done in our work. The spread activation proposed in our work can also be used to provide rankings for associations among concepts in the instances graph. Both Path Length Weight (in the activation decay and path constraints) and Context Weight (through the configuration of the spread activation) are also present in our work. We envision that both Subsumption Weight and Trust Weight ideas can be successfully integrated in our framework to provide even better results. Some heuristics presented in their work are also present in our work through the constraints presented in Section 3.2.2.

8. CONCLUSIONS

The semantic search proposed in this paper combines traditional search engine techniques together with ontology based information retrieval. It proved to be very successful in two different applications tested. In both applications, one of the concepts in the ontology had a lot of textual information (the "Publication" concept in the D.I. website application and the

"Painting" concept in the Portinari application). We believe that the proposed semantic search is especially useful in all applications domains where this occurs. The search will make it possible to link concepts hidden in the keyword description of these textually rich attributes, with semantic meaningful ontology instances that are present in the knowledge base.

One of the problems in the spread activation algorithm proposed is that there is no semantic interpretation of the activation value flowing through the network. Some inferences made in the process are not true and should not be considered. The implementation of a Relevance Feedback algorithm seems to be the best alternative to tell the system how well it is doing. This would greatly help the knowledge engineer, since after some time the system would learn the best configuration and the preferable paths in a particular domain.

Since the system was implemented as a framework, it is straightforward to extend it. The first set of expansions will seek to improve existing functionality by experimenting different weight mapping formulas and spread activation configurations. A second set of expansions will add new features to the system, such as incorporating subsumption weight, context sensitive weight mapping, relevance feedback, etc.

We are also studying alternative ways of evaluating the semantic search proposed. In particular, we are trying to devise measures that would make it possible to compare our search with other existing semantic searches in a quantitative way. As mentioned before, we also want to test the system in applications with larger ontologies to evaluate how scalable it is.

Another area we wish to further investigate and test is the use of hybrid queries in our semantic searcher. We consider a hybrid query to be one in which the user can mix keywords with instances of concepts in the knowledge base. For example, the user could propose a query like *oohdm & (Professor: "Daniel Schwabe")*, where part of the query is specified in terms of keywords ("oohdm") and part is specified in terms of concepts instances (the instance "Daniel Schwabe" of type Professor). This functionality is already implemented in our system but has not been extensively tested.

We envision other applications where the spread activation system proposed here would be very useful. We are doing some experiments in using the system for the automatic recommendation of new relations instances based on the analysis of the existing knowledge base. The spread activation can propose a set of concepts which seem to be strongly connected to a given concept even though no explicit relation between the concepts exists in the knowledge base. This would help the users of an application in the task of filling in new instances to be added to the knowledge base. This functionality is also very useful in identifying possible inconsistencies in the knowledge base since it can identify relations that have a great chance of existing even though they are not explicit in the knowledge base.

Acknowledgement. The research presented in this paper was partly funded by scholarships from PUC-Rio and CNPq, and research grants from CNPq and FAPERJ. We also want to thank both LES/LAC and TecWeb laboratories for providing the necessary infra-structure for developing this work.

9. REFERENCES

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

[2] Chen, H., and Ng, T. An Algorithmic Approach to Concept Exploration in a Large Knowledge Network (Automatic Thesaurus Consultation); Symbolic Branch-and-Bound vs. Connectionist Hopfield Net Activation. Journal of the American Society for Information Science 46(5):348-369, 1995.

[3] Cohen, P., and Kjeldsen, R. Information Retrieval by Constrained Spreading Activation on Semantic Networks. Information Processing and Management, 23(4):255-268, 1987.

[4] Crestani, F. Application of Spreading Activation Techniques in Information Retrieval. Artificial Intelligence Review, 11(6): 453-482, 1997.

[5] Davies, J., Weeks, R., and Krohn, U. QuizRDF: Search Technology for the Semantic Web. WWW2002 workshop on RDF & Semantic Web Applications, Proc. WWW2002, Hawaii, USA, 2002.

[6] Fayad, M., Schmidt, D., and Johnson, R. Building Application Frameworks. Wiley Computer Publishing, 1999.

[7] Froogle. http://froogle.google.com

[8] Guha, R., and McCool, R. Tap: Towards a web of data. http://tap.stanford.edu/.

[9] Guha, R., McCool, R., and Miller, E. Semantic Search. Proceedings of the WWW2003, Budapest, 2003.

[10] Lucene Search Engine. http://jakarta.apache.org/lucene

[11] O'Hara, K., Alani, H., and Shadbolt, N. Identifying Communities of Practices: Analyzing Ontologies as Networks to Support Community Recognition, IFIP-WCC 2002, Montreal, 2002, Kluwer.

[12] Peat, H., and Willet, P. The limitations of term co-occurrence data from query expansion in document retrieval systems. Journal of the American Society for Information Science, 42(5), 378-383, 1991.

[13] Portinari Project Website. http://www.portinari.org.br

[14] PUC-Rio Informatics Dept. http://www.inf.puc-rio.br.

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

[16] Srikant, R., and Agrawal, R. Mining generalized association rules. Proceedings of VLDB '95, pp. 407-419, 1995.

[17] Stojanovic, N., Struder R., and Stojanovic, L. An Approach for the Ranking of Query Results in the Semantic Web. Proc. of ISWC '03 (Sanibel Island, FL, October 2003), Springer-Verlag, 500-516, 2003.

[18] Yates, B., and Neto, B. Modern Information Retrieval. ACM Press, New York, USA, 1999.

[19] Zobel, J., and Moffat, A. Exploring the similarity space. ACM SIGIR Forum 32(1):18-34, Spring, 1998.