An Approach for the Ranking of Query Results in the Semantic Web

Nenad Stojanovic[1], Rudi Studer[1,2,3], and Ljiljana Stojanovic[2]

[1] Institute AIFB, University of Karlsruhe, 76128 Karlsruhe, Germany
{nst,rst}@aifb.uni-karlsruhe.de
[2] FZI - Research Center for Information Technology at the University of Karlsruhe, 76131 Karlsruhe, Germany
Ljiljana.Stojanovic@fzi.de
[3] Ontoprise GmbH, Amalienbadstraße 36, 76227 Karlsruhe, Germany

Abstract. One of the vital problems in the searching for information is the ranking of the retrieved results, because users make typically very short queries (2-3 terms) and tend to consider only the first ten results. In traditional IR approaches the relevance of the results is determined only by analysing the underlying information repository (content and hyperlink structure), which leads to the weak relevance model. On the other hand, in the Semantic Web the querying process is supported by an ontology such that other important sources for determining the relevance of results can be considered: the structure of the underlying domain and the characteristics of the searching process.
In this paper we present a novel approach for determining relevance in ontology-based searching for information, which exploits the "full potential" of the semantics of such a semantically-based link structure. We present several analyses about how a Semantic Web querying mechanism can benefit of using our ranking approach.

1 Introduction

One of the main goals of the current Web is information sharing, i.e. the possibility to access information without considering a particular platform, language, protocol. It was the business need behind the current Web [1]. Consequently, the focus of the Web research in the last ten years was on the standards for ensuring that Web information resources can be accessed. The development of the Semantic Web and the layering in the Semantic Web architecture should enable more efficient inclusion of machine agents in the process of searching for information resources. Moreover, the higher layers in the architecture should enable the explanation of how a piece of information was found and the reason for trusting the information. Even the emerging industrial interest in the Web Services has to rely on the Semantic Web functionalities to find a service efficiently (semantically) [2].

In this paper, we elaborate on the benefits to be expected in searching for information, when exploiting Semantic Web technologies. More precisely, based on the experiences from Web information retrieval (IR), we analyse the challenges for Semantic Web IR and develop an approach that exploits the explicitly shared

D. Fensel et al. (Eds.): ISWC 2003, LNCS 2870, pp. 500–516, 2003.

semantics of the information for the more efficient searching in the Semantic Web. Besides the improvements in the precision and recall [3], we see two other very important and practical benefits of using an ontology in the searching for information: better possibilities for ranking and clustering the retrieved information. We present a novel approach for ranking the results of ontology-based searching and show its implications on Semantic Web IR. The approach combines the characteristics of the inferencing process and the content of the information repository used in searching. It relies on our previous work on ontology-based inferencing [4] and ontology-based information retrieval [5], as well as on our current efforts in developing a Semantic Web infrastructure [6] and research in the Semantic Web Mining [7]. The approach can be very easily extended with any additional information related to the search process, e.g. the frequency of usage of ontology-based information. We have implemented the approach in the Ontobroker inference engine (ontobroker.semanticweb.org) and incorporated it in the Semantic Portal of our Institute. We present an evaluation study about the performance of this approach.

The paper is structured as follows: In the second section we introduce the terminology we use in this paper. In section 3 we present our ontology-based ranking schema and discuss its advantages for searching in the Semantic Web, whereas in section 4 we give implementation details. Results of our evaluation study are presented in section 5. Related work is presented in section 7 and section 8 contains some concluding remarks.

2 Background

In this section we give the basic terminology we use in this paper. Due to lack of space some definition of lower importance for understanding this paper are omitted.

Definition 1: Ontology
A core ontology is a structure $O := (C, \leq_c, R, \sigma, \leq_r)$ consisting of

- two disjoint sets C and R whose elements are called concept identifiers and relation identifiers, resp.,
- a partial order $\leq_c$ on C, called concept hierarchy or taxonomy (without cycles)
- a function $\sigma : R \rightarrow C^+$, called signature
- a partial order $\leq_r$ on R called relation hierarchy, where $r_1 \leq_r r_2$ implies $|\sigma(r_1)| = |\sigma(r_2)|$ and $\pi_i(\sigma(r_1)) \leq_c \pi_i(\sigma(r_2))$ for each $1 \leq i \leq |\sigma(r_1)|$

Often we call concept identifiers and relation identifiers concepts and relations, respectively, for the sake of simplicity.

Definition 2: Domain and Range
For a relation $r \in R$ with $|\sigma(r)| = 2$, we define its domain and its range by $dom(r) := \pi_1(\sigma(r))$ and $range(r) := \pi_2(\sigma(r))$.

Definition 3: Axioms (Rules)
Let L be a logical language and $L(O)$ be the set of statements in L w.r.t. an ontology O. An L-axiom system for an ontology $O := (C, \leq_c, R, \sigma, \leq_r)$ is a pair $A := (AI, \alpha)$, where

- AI is a set whose elements are called axiom identifiers and
- $\alpha : AI \rightarrow L(O)$ is a mapping.

The elements of $\alpha(AI)$ are called axioms or rules.

An ontology with L-axioms is a pair (O, A), where O is an ontology and A is an L-axiom system for O.

Definition 4: Knowledge Base
A Knowledge Base is a structure $KB := (C_{KB}, R_{KB}, I, l_c, l_r)$ consisting of

- two disjoint sets C_{KB} and R_{KB}
- a set I whose elements are called instance identifiers (or instances or objects in brief)
- a function $l_C : C_{KB} \rightarrow I$ called concept instantiation
- a function $l_r : R_{KB} \rightarrow I^+$ called relation instantiation

A relation instance can be depicted as $r(I_1, I_2, ..., I_n)$, where $r \in R_{KB}, I_i \in I$. r is called a predicate and I_i is called a term.

Definition 5: Query
A (conjunctive) query is of the form or can be rewritten into the form:

$$forall\ \overline{X}\ \ \overline{P}(\overline{X}, \overline{k}) \text{ and } not(\overline{N}(\overline{X}, \overline{k}))$$

with $\overline{X}$ being a vector of variables $(X_1, ..., X_n)$, $\overline{k}$ being a vector of constants (concept instances), $\overline{P}$ being a vector of conjoined predicates (relations) and $\overline{N}$ a vector of disjoined predicates (relations). A query can be viewed as an axiom without a head.

For example, for the query "forall x worksIn(x, KM) and researchIn(x, KMsystems)" we have

$$\overline{X} := (x),\ k := (KM, KMsystems),\ \overline{P} := (P_1, P_2),\ P_1(u,v,w) := worksIn(u,v),$$
$$P_2(u,v,w) := researchIn(u,w).$$

Note: In this paper we consider only conjunctive queries. Since a disjunctive query can be represented as a disjunction of several conjunctive queries our approach can be easily extended to disjunctive queries.

Definition 6: Answers (results) of a query
Let Ω be the set of all relation instances which can be proven in the given ontology (this set can be obtained by the materialisation of all rules).

For a query Q " $forall\ \overline{X}\ \ \overline{P}(\overline{X}, \overline{k})$ and $not(\overline{N}(\overline{X}, \overline{k}))$ " an answer is an element (tuple) in the set $R(Q) = \{\overline{X}\} = \{(x_1, x_2, ...x_n)\}$, such that $\overline{P}(\overline{X}, \overline{k})$ and $not(\overline{N}(\overline{X}, \overline{k}))$ is provable, i.e. each of the relation instances $r(x_1, x_2, ..., x_n, k_1, ...k_l),\ r \in \overline{P}$ exists in the set Ω and each of $l(x_1, x_2, ..., x_n, k_1, ...k_l), l \in \overline{N}$ does not exist in Ω.

3 The Ranking in the Semantic Web

3.1 Introduction

There are two crucial differences between Web IR and searching for information in the Semantic Web:

- Instead of searching for web documents, an agent in the Semantic Web searches for resources which satisfy some formally defined conditions, i.e. all resources which are retrieved are relevant for the given query. Consequently, the criteria precision and recall, which are very popular for the estimation of the quality of the Web IR, are not so much useful for the resource retrieval in the Semantic Web.
- Searching is treated as ontology-based inferencing, which enables retrieval of information that is not explicitly stored in the information repository. By using the background knowledge implied by the underlying domain ontology, some new, implicitly stated, statements can be inferred in the querying process. In that way, although all retrieved results are relevant for a given query, their inferencing processes can be different and used for ranking the results.

Both differences emphasize the role of ranking the results that are retrieved for a query in Semantic Web. Even when machine agents process the list of results automatically, it is not realistic to assume that they will inspect the whole list of thousands of retrieved results. Moreover, in real applications, e.g. skill management, the retrieved resources can be people, e.g. experts, and just several of them can be contacted.

The ranking of the results of a search process in the Semantic Web should benefit from all possible advantages implied by using the conceptual model of a domain, i.e. the domain ontology, in describing the information on the Web. The strength of an ontology lies in the formal and explicit specification of the constraints which exist between domain entities (an entity can be a concept or relation). One of the most powerful mechanisms for describing constraints between entities in a domain are rules (see Definition 3). The rules are part of the knowledge about a domain and are extensively used for deriving results of an ontology-based query, i.e. in the inferencing process. As we already mentioned in the previous section, the way in which a result of a query is derived can help in ranking the results of that query. Subsequently, we present such a ranking approach, which combines characteristics of the inferencing process and the content of the information repository used in searching. Due to the strong dependence on the usage of ontology background, we call it *ontology-based ranking*.

The importance of the rules for the ontologies on the web is already recognised in the Semantic Web community and several initiatives for representing rules in the RDF compatible format are already started, e.g. RuleML [8]. However, since the current W3C (proposal) standard for representing ontologies on the Web, OWL [9], does not cover the rules (yet), we have to use an ontology representation formalism which enables dealing with rules. Since our approach is partially implemented in the Ontobroker system [4], which relies on the F-Logic [10], in the rest of the paper we will use F-Logic notation for representing ontologies. However, the generality of our approach remains.

3.2 Motivating Example

In order to make the discussion more understandable, we will introduce and motivate the requirement for a ranking approach with an example, firstly. The example is taken from our evaluation study we present in section 5. In the rest of the text we will refer to it as the *institute example*.

Let us assume the following ontology:

```
1: Project::Object[hasTopic =>> Topic].
2: Lecture::Object[hasTopic =>> Topic].
3: Topic::Object[subtopicOf =>> Topic].
4: Researcher::Object.                                                    (1)
5: Professor:: Researcher.
6: PhDStudent:: Researcher.
7: Researcher [worksIn =>> Project; researchIn =>> Topic; teaches =>> Lecture].
8: FORALL X,Y,Z   Z: Researcher [researchIn –>>Y] <– Z[worksIn –>>X] and
        X:Project[hasTopic –>>Y].
9: FORALL X,Y,Z   Z: Researcher [researchIn –>>Y] <– Z[teaches –>>X] and
        X:Lecture[hasTopic –>>Y].
10:FORALL X,Y,Z   Z:Project[hasTopic –>>Y] <– X:Topic[subtopicOf –>>Y] and
        Z[hasTopic –>>X].
```

To give an intuition of the semantic of the F-Logic statements, in line 1, one finds a concept definition for a `Project` being an `Object` with a relation `hasTopic`. The range of the relation is restricted to `Topic`. Ontology axioms as the one given in line 8 (1) use this syntax to describe regularities. Line 8 states that if a `Researcher` *Z* works in a `Project` *X* and *X* has topic *Y,* then *Z* does research in *Y*. Let us further assume the following knowledge base:

```
11: rst:Professor.
12: gst:Professor.
13: meh:PhdStudent.
14: nst:PhdStudent.                                                       (2)
15: ysu:PhdStudent.
16: KM:Topic.
17: TextMining:Topic.
18: DataMining:Topic.
19: rst[worksIn->>OntoWeb; worksIn ->>DotCom; worksIn ->>OTK].
20: ysu[worksIn ->>OTK].
21: rst[teaches ->>KnowledgeManagement].
22: nst[teaches ->>KnowledgeManagement].
23: gst[teaches ->>KnowledgeManagement;worksIn->>DotCom].
24: gst[teaches ->>InfoA;teaches ->>InfoB].
25: gst[worksIn ->>OntoWeb; worksIn ->>DotCom; worksIn ->>SWAP].
26: meh[worksIn ->>SWAP;teaches->>InfoB].
27: OntoWeb:Project[hasTopic->>KM;hasTopic->>TextMining; hasTopic->>DataMining].
28: WonderWeb:Project.
29: OTK:Project[hasTopic->>KM].
30: DotCom:Project[hasTopic->>KM].
31: SWAP:Project[hasTopic->>TextMining].
32: TextMining[subtopicOf->>KM].
33: DataMining[subtopicOf->>KM].
34: KnowledgeManagement:Lecture[hasTopic->>KM].
35: InfoA:Lecture.
36: InfoB:Lecture.
```

Definitions of instances in the knowledge base are syntactically very similar to the concept definition in F-Logic. In line 11 the instance `rst` of the concept `Professor` is defined. Furthermore, in line 20, the relation `worksIn` is instantiated between `ysu` and `OTK`. Similarly, in line 29 it is stated that `OTK` is a `Project` related to the topic `KM`.

Now, an F-Logic query may ask for all people who research in KM by:

```
Qa: FORALL Y <- Y[researchIn ->> KM].
```
(3)

which may result in the set $R(Qa) = \{$(rst), (sst), (ysu), (meh), (gst), (nst)$\}$.

By using the associations: $\overline{X} :=$ *(Y)*, $\overline{k} :=$ ("KM"), $\overline{P} :=$ (P_1), $P_1(Y,"KM") :=$ Y[researchIn ->>"KM"], the query (3) can be represented in the formal manner (Definition 5) introduced in section 2.

Obviously, all answers are correct with regard to the given knowledge base and ontology, but the question is how these answers are related to each other. Let us consider the intuitive assumptions about the relevance of the different results in the following cases:

1) ysu vs. gst
   - ysu works in just one project and that project is about KM
   - gst works in three projects but just one of them is about KM

     => ysu is more relevant for KM than gst.

2) rst vs. ysu
   - ysu works in just one project and that project is about KM
   - rst works in three projects, all related to KM

     => rst is more relevant for KM than ysu.

3) gst vs. nst
   - gst is a Professor and gives a lecture about KM
   - nst is a PhDStudent and gives the same lecture as gst

     => gst is more relevant for KM than nst

4) nst vs. meh
   - nst works in just one project and that project is about KM
   - meh works in just one project and that project is about TextMining, which is a special part of KM research

     => nst is more relevant for KM than meh

5) ysu vs. nst
   - ysu works in just one project and that project is about KM
   - nst works in just one project and that project is about KM

     => the relevance of ysu and nst is the same,

   Alternatively, we can use more semantic about these relations:
   - ysu works in just one project and that project is about KM and he works alone
   - nst works in just one project and that project is about KM and there are two other persons who support the research in that project

It could be concluded that ysu is more relevant than nst, but the following arguments might be applied:
- ysu works for the whole project => ysu puts a lot of efforts in the project
- nst works with two other persons in the project => nst benefits from the collaboration with two other persons

  Therefore, the relevance depends on additional factors, e.g. what is the proportional active participation of all three members in that project.

It is clear that a powerful, but flexible ranking schema is needed, in order to fulfil all requirements presented in this example. This is the goal of our ranking approach. In the discussion of the approach (cf. the end of the next section) we explain the solutions for the above mentioned requirements.

3.3 The Ontology-Based Ranking Schema

In this subsection we give the theoretical framework for calculating the relevance of the returned results for a query, which resolves the above mentioned cases.

Although a query returns the set of concept instances as an answer, the relevance of these answers is defined on the level of the relation instances. The reason is that the concept instance is treated as the identifier of an object (e.g. `rst`), whereas the relation instance (e.g. `rst[researchIn->>KM]`) represents the property of that object whose relevance for the query can be determined. It means that the relevance of an answer $a = \overline{X}_1 = (x_1, x_2, ...x_n)$ for the query Q will be calculated on the substitution of this answer in the query Q, i.e. by considering the $\overline{P}(\overline{X}_1, \overline{k})$: $P_1(\overline{X}_1, \overline{k}), P_2(\overline{X}_1, \overline{k}), ..., P_k(\overline{X}_1, \overline{k})$ - the set of *returned relation instances*, depicted as $\Lambda(a)$, for an answer a. It is clear that $\forall a$, $\Lambda(a) \supset \Omega$ (for the used terminology see section 2).

For the given example and query (3) follows:

```
Λ(rst)={rst[researchIn->>KM]}, Λ(sst)={sst[researchIn->>KM]}
Λ(ysu)={ysu[researchIn->>KM]}, Λ(gst)={gst[researchIn->>KM]}
Λ(nst)={nst[researchIn->>KM]}, Λ(meh)={meh[researchIn->>KM]}
```

Since the returned relation instances for each answer can be proven in the ontology, the traditional definition of the relevance via a similarity function between returned relation instances [11] is useless (similarities between two arbitrary answers are equal).

There are two differences between *returned relation instances* for answers to a query:

(i) the specificity of the instantiation of the ontology (content of the knowledge base)
e.g. for the domain presented in the motivating example, for a query about researchers in `KM`, a useful information for ranking can be that a researcher works in three projects about `KM` and gives two lectures regarding `KM`

(ii) the inferencing process in which an answer is implied - the derivation tree of an answer (a returned relation instance)
e.g. the query for a researcher who researches in `KM` will return the researcher who researches in the `DataMining` as well, since `DataMining` is a subtopic of `KM`.

Therefore, we introduce the relevance function $\sigma: \Omega \rightarrow \Re$ that computes the relevance of a relation instance returned in the querying process, based on analysing (i) the knowledge base (criterion i) and the inferencing process (criterion ii). The relevance of an answer a, $\rho(a)$, $a \in R(Q)$, is the geometrical mean of the relevancies of the returned relation instances for that answer, i.e. $\rho(a) = \sqrt[|\Lambda(a)|]{\prod_{x \in \Lambda(a)} \sigma(x)}$, where $|S|$ denotes the cardinality of a given set S. Finally, the ranking of the answers for a query is achieved by ordering them according to their relevance. In the following two subsections we give in detail the calculation of the above mentioned relevancies.

3.3.1 The Content of the Knowledge Base

In this subsection we explain the calculation of the relevance of a relation instance for the user's query.

Definition 7: The ambiguity of a term in a relation instance
The ambiguity of a term (a concept instance) in a relation instance is defined as the number of interpretations of the given relation instance with respect to that term (i.e. when all other terms in the relation instance, except the considered term, are substituted with a variable).

$$Amb(I_j, r(I_1, I_2, ..., I_j, ..., I_n)) = \left|\{x, y, ..., w \mid \exists x, y, ...w \quad r(x, ..., I_j, ..., w)\}\right| \tag{4}$$

For example, $Amb(gst, worksIn(gst, OntoWeb)) = \left|\{x \mid \exists x \quad worksIn(gst, x)\}\right| = 3$ (regarding motivating example, person `gst` works in three projects `OntoWeb, DotCom, SWAP`).

Definition 8: The specificity/relevance of a relation instance
The specificity of a relational instance is the reciprocal value of the ambiguity of each of its terms (concept instances). It is calculated as:

$$Spec(r(I_1, I_2, ..., I_n)) = \frac{1}{Amb(I_1, r(I_1, I_2, ..., I_n))} \cdot ... \cdot \frac{1}{Amb(I_n, r(I_1, I_2, ..., I_n))} \tag{5}$$

For example, regarding the situation presented in (1), the specificity of the instance *workIn(gst, OntoWeb)* is as follows: $Spec(worksIn(gst, OntoWeb)) =$

$$\frac{1}{Amb(gst, worksIn(gst, OntoWeb))} \cdot \frac{1}{Amb(OntoWeb, worksIn(gst, OntoWeb))} = \frac{1}{3} \cdot \frac{1}{2} = \frac{1}{6} \tag{6}$$

The specificity is maximal (i.e. = 1) when none of the terms from the relational instance can be found in any other relation instance of the same type[1]. For example, the relation instance `teaches(meh, InfoB)` (in F-Logic `meh[teaches->>InfoB].`) has the maximal specificity, because there is only one relation instance for the relation *`teaches`* which contains either `meh` or `InfoB`.

The specificity of a relation instance can be interpreted as the measure of its relevance for the user's query. When the specificity of a relation instance is higher, then the relevance is higher as well.

Hence, the relevance is calculated as: $Rel(r(I_1, I_2, ..., I_n)) = Spec(r(I_1, I_2, ..., I_n))$ (7)

When considering the case 5) from section 3.2, the treatment of factors which determine relevance is task-dependent. The formula (5) can be adapted to such task-dependent customisation by changing the reciprocity-effect of the ambiguity of a term on the relevance of the whole relation instance. For example, when the relevance of a participant of a project for the research related to that project increases with the number of participants of the project, then the statement (6) can be changed into (note that we have two participants in `OntoWeb`): $Spec(worksIn(gst, OntoWeb)) =$

$$\frac{1}{Amb(gst, worksIn(gst, OntoWeb))} \cdot Amb(OntoWeb, worksIn(gst, OntoWeb)) = \frac{1}{3} \cdot 2 = \frac{2}{3}$$

Our approach supports this kind of parameterised calculation of the relevance.

3.3.2 The Characteristics of the Inferencing Process

As showed in the motivating example, in querying an ontology the background knowledge about the domain is used for inferring new, implicitly stated facts in the knowledge base. Therefore, the query results might be extended by inference, which

---

1 A relation type is defined by the relation that is instantiated in the relation instance.

uses rules from the domain ontology, i.e. the query process includes ontology-based inferencing. We omit here the detailed description of the different types of inferencing processes (evaluations) [12], but present instead the common inferencing structure which can be used for the proof of the returned results. Such a proof structure enables the reconstruction of the inferencing steps for a query's result and consequently the calculation of its relevance. This structure is described in the next three definitions.

Definition 9: AND-OR tree

An AND-OR tree is a tree structure, defined as the set T = $\{root, N, v\_and, v\_or\}$, where *root* is the root of the tree, *N* is the set of nodes in the tree, *v_and* and *v_or* are (irreflexive, anti-symmetric) relations between nodes (they define links in the tree and are called parent-child relations in the text), *v_and*: $N \rightarrow N$ (such a link is called an and_link and the node is called an and_connector), *v_or*: $N \rightarrow N^*$ (such a link is called an or_link and the node is called an or_connector).

Definition 10: Derivation tree of a query

Given an ontology O with axioms, the derivation tree of the query Q is an AND-OR tree whose root plays the role of an or_connector between the derivation trees of each result (resulting relation instance) *R*(Q) for the query Q, i.e.

$DTree(Q) = \{root, N, v\_and, v\_or\}$, $N = \bigcup_{r \in \Lambda(a) \wedge a \in R(Q)} DTree(r)$, *v_or*: root→N , *v_and* = { }.

Definition 11: Derivation tree of a relation instance (creating)

The derivation tree of a relation instance $r(I_1, I_2, ..., I_n)$ is defined as follows:

- Every relation instantiation *l* from KB is a derivation tree for itself; a single node with label *l*.
- Let *A* be the set of axioms from the given ontology O, whose heads contain the atomic formula which can be unified with the relation *r*.
- Let A = $p :- q_1, ..., q_n$, be an axiom from *A*, let d_i, $1 \le i \le n$, be relation instances with derivation trees T_i, let θ be the mgu[2] of $(q_1, ..., q_n)$ and $(d_1, ..., d_n)$. Then the following is a derivation tree for p[θ] (relation instance): the root is a node labelled with p[θ] and each T_i, $1 \le i \le n$, is a child of the root. The root plays an and_connector role.
- The derivation tree for *r* is the tree with the root which plays the role of an or_connector between the derivation trees for all axioms from *A*. It is labelled with p[θ]*

Fig. 1 depicts a derivation tree, created according to the definitions 9, 10 and 11.

---

[2] A substitution is a mapping from the set of variables of the language under consideration to the set of terms. Two terms t1 and t2 are said to be unifiable if there is a substitution σ such that t1[σ] = t2[σ]; σ is said to be a unifier of t1 and t2. Note that if two terms have a unifier, they have a most general unifier (mgu) that is unique up to renaming of variables.

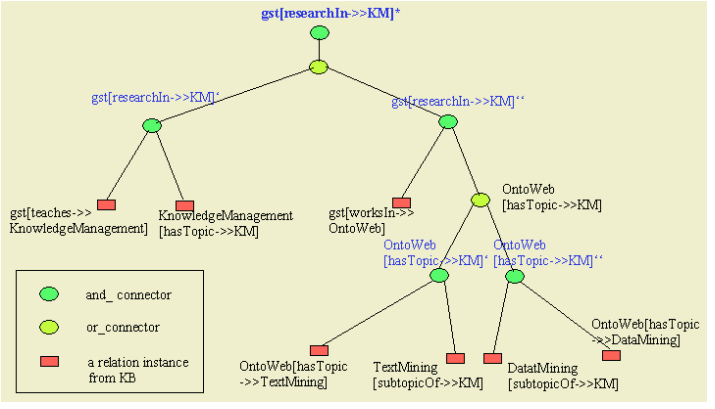

Fig. 1. The derivation tree for the relation instance `gst[researchIn->>KM]`, regarding the motivating example. The tree is generated according to the definitions 9, 10 and 11.

It is clear that a derivation tree represents the manner in which a result was inferred. The root is the disjunction of all results. Each node in a tree is a relation instance. Each or_connector node connects various ways in which a relation instance is derived, whereas an and_connector node defines a way in which the relation instance is derived. For a relation instance there are several and_connector nodes (the number is equal to the number of rules, whose head parts unify with that relation instance) and an or_connector, which connects those and _connector nodes.

For each and_connector node it is possible to define the query specificity as a measure of the relevance of that relation instance (using formula (5)). From the definition of the derivation tree it is clear that the relevance of a node depends on the relevance of its children's nodes. A relevance is propagated in two ways depending on the type of nodes:

- an and_connector multiplies the relevancies of its children (because each child is a part of a rule that infers the parent)
- an or_connector sums the relevances of its children (because children make an impact on the parent independently)

Such propagation corresponds to the propagation of the signals in a semantic network, so called node activation sequences [13].

The following formulas describe the processing done by a node:

(i) for an and_connector $node_i$: $Rel\_and(node_i) = Rel(node_i) \cdot \sum_{v\_or(node_i, node_j)} Rel\_or(node_j)$ (8)

for the root node *root*, we set $Rel(root) = 1$.

(ii) for an or_connector $node_i$: $Rel\_or(node_i) = \prod_{v\_and(node_i, node_l)} Rel\_and(node_l)$ (9)

(iii) for a leaf $node_i$: $Rel\_and(node_i) = Rel(node_i)$ (10)

Each node is in the form $r(I_i, I_2, ..., I_n)$, *Rel(node)* is the initial relevance, defined with formula (7), *Rel_and(node)* is the calculated relevance of the relation instance

corresponding to *node*. Therefore, the relevance of a returned relation instance *r* for the answer *a*, i.e. $r \in \Lambda(a)$, is calculated as follows: $\sigma(r) = Rel\_and(r)$.

Finally, the relevance of the answer *a* is $\rho(a) = |\Lambda(a)| \sqrt{\prod_{r \in \Lambda(a)} Rel\_and(r)}$. (11)

Discussion:

(i) The ranking measure described with (11) deals with both factors for determining the relevance we mentioned in previous section (i.e. regarding content and inference)

(ii) In formula (5) we treat the inverse impact of the number of relations of the same type *r* as follows: when an instance is in several relations of the same type, then its relevance for that relation type is split into all of them. This enables resolving of the case 2) from section 3.2. However, when each of these relations are relevant for the given query, according to summing in formula (8) the instance has an impact "without losses" (= 1) on this relation. This is important for resolving case 1) from section 3.2.

(iii) Since for each transitive relation there is a rule which enables inferencing over the subconcepts, the derivation trees of two subconcepts, which are placed in the different depth in the hierarchy of a term, are different. Consequently, their relevancies (formula (11)) are different, i.e. the subconcepts placed deeper in the hierarchy are less relevant for the queries regarding the root concept. It enables resolving of the case 3) and 4) from section 3.2.

(iv) The case 5) form section 3.2 was treated in the discussion regarding Definition 8.

(v) The formula (5) for calculating the specificity of a relation instance is general enough to model other interpretations of the relevance (see the discussion in the case 5 in section 3.2) as well as to incorporate other factors which can determine the relevance, e.g. usage information. We see the calculation of the relevance as a task/problem-sensitive decision and we plan to develop task-oriented strategies for calculating relevance. For example, whether the number of the instances which are in the same relation should increase or decrease the relevance is a problem-sensitive decision and we assume that a user wants to have opportunity to set such an indicator for each relation, otherwise he/she will use default option.

4 Implementation

4.1 Ontobroker and Its Explanation Facility

We implemented the presented ranking approach in the Semantic Portal of our Institute. It is an ontology-based web application, which serves as the test-bed for our research related to ontologies and Semantic Web. The Semantic Portal (SEAL) [5] is an ontology-based application, which provides a "single-click" access to almost all information related to the organisation, people, research and projects of our Institute. It is widely used by our research and administrative staff as well as by our students.

One of the most usable features is the possibility to search for people, research areas and projects on the semantic basis, i.e. using the corresponding Institute Ontology. The hierarchy of research areas is especially comprehensive – more than 130 concepts. The example from section 3 is a part of it. The portal provides a very user-friendly interface, which enables formation of arbitrary queries using entities from the underlying ontology. The search is performed as an inference through metadata, which are crawled from the portal pages. As the inference mechanism we use the Ontobroker system [4], a deductive, object-oriented database system operating either in the main memory or on a relational database (via JDBC). Ontobroker uses the bottom-up fix-point evaluation procedure. It allows general recursion and negation is allowed in the clause body. The native logical language of the Ontobroker is F-Logic. More details about Ontobroker can be found in [4].

The latest version of Ontobroker (www.ontoprise.com) has a powerful explanation facility, which tracks the evaluation of the rules and the substitutions applied in them. The ranking approach is based on processing this explanation file. Out of it, the derivation tree for a query, according to the definitions 9-11, is generated. Since Ontobroker generates explanations in the form of F-Logic statements, the processing of the explanation file is logic-based. Indeed, we define a small ontology for processing explanation statements, including the set of rules which enables the generation of the desired AND-OR tree. Such an approach enables a very efficient customisation of the explanation process. The recursive rules and the negation do not impose any problems for the processing of the explanation file. The explanation file explains the original logic program, i.e. no optimisation technique (e.g. Magic set) affects the explanation process. Due to the lack of space we omit here more details.

The ranking approach is developed as an additional module for the Ontobroker system. The ranking module takes the list of results for the query and the explanation file as inputs. The output is the ranked list of results. Case studies show that the time delay introduced in the answering process is not significant.

4.2 Evaluation of Ranking Performances

In our previous work [5] we developed a module for ranking the results retrieved by Ontobroker, based on the calculation the similarity between terms in a hierarchy (i.e. for any transitive relation). The measure is based on the assumption, that the similarity between two objects (concepts or instances) may be computed by considering their relative place in a common hierarchy H. H may, but need not be a taxonomy. For instance, in our test domain we have a categorization of research topics – *H(subtopicOf)*, which is not a taxonomy. The similarity between two objects in a hierarchy is calculated using the so-called *Object Match* measure. In the following we describe very briefly this measure.

For each object we define *Upwards cotopy* (UC) as the number of objects on the paths between the given object and the root of the given hierarchy H. The *Object Match* (OM) between two objects, O_1, O_2, is defined as $OM(O_1, O_2, H) = \frac{UC(O_1, H) \cap UC(O_2, H)}{UC(O_1, H) \cup UC(O_2, H)}$. Basically, OM reaches 1 when two objects coincide; it degrades to the extent to which the discrepancy between intersections and

unions increases (an *OM* between concepts that do not share common super-concepts yields value 0). More details can be found in [5].

The problem in such a calculation of the relevance is that a part of the domain model (i.e. the domain axioms) is not treated at all. It leads to a weak relevance model, which takes into account only the hierarchy of relations in the ontology. For example, the difference of the relevance for the research area `KM` between a `Researcher`, who is the leader of a `KM`-project and another `Researcher`, who is "only" a participant in a `KM`-project, cannot be expressed using our former ranking approach. For real-world applications we have developed using the Ontobroker, it was a very important issue, especially in the skill-management domain.

By involving more semantics about the domain (i.e. axioms) in calculating the relevance, we expected that the new approach for ranking will outperform the old one. Therefore, we set up an evaluation study in order to prove this claim.

Due to the subjective nature of relevance, it is very difficult to evaluate the performance of a ranking algorithm. Here we used a modification of the method for interactive comparing of two ranking algorithms, proposed in [14].

Our experiment is set up as follows: For a query, the set of predefined answers is determined. These answers are processed by both ranking algorithms. The returned rankings are mixed, so that at any point the top l results of the combined ranking contain the top ka and kb rankings from A and B, $\|ka - kb\| \leq 1$. The combined ranking for the given query is presented to the user, and the user is asked to select $l/2$ of the most relevant results. We calculate the number of these top $l/2$ results chosen from each of the two ranking algorithms, the so-called *top_results_ratio*, as:

$$top\_results\_ratio(X) = \frac{top(X)}{Num(X)},$$

where $top(X)$ is the number of results from the ranking X in the selected top $l/2$ results and $Num(X)$ is the total number of results from the ranking X presented to the user, i.e. for X=A, $Num(X)= ka$ and for X=B, $Num(X)= kb$. The ranking with higher *top_results_ratio* has the better ranking schema.

In the original method [14], the ranks are calculated in a different way: users just click on one or two results, these clicks are recorded and afterwards the clickstream data are analysed.

For this evaluation step we used the data from the Semantic Portal of our Institute.

For the experiment we selected the answers of 20 predefined queries and performed the ranking of these answers by both algorithms. The queries were about researchers who research in different research area, e.g. in the F-logic: `FORALL X, X[reserachIn->>"name_of_research_area"]`. The number of answers varies from 8 to 15. We set the values l, ka and kb to 16, 8 and 8, respectively. We used ten subject experts in the evaluation. They did not have previous knowledge about the system.

In order to evaluate the additional cost of our algorithm, we measured the processing time for all algorithms. The results of the evaluation study are given in table 1. In the table wins/total is the ratio of the number of trials in which the corresponding algorithm won and the total number of trials (=200).

Table 1. The results of the evaluation.

Algorithm	**Based only on hierarchy**	**Based on the full domain model**
Ranking (wins/total)	8/200	192/200
average process. time	703 ms	850 ms

Discussion: The experiment proved that using more semantics about the domain in a ranking algorithm improves the ranking drastically, without increasing the processing time significantly. The 8 losses, we accounted in the evaluation, were for the sets of answers which do not require a lot of semantics to be properly ranked. Moreover, the differences in results in these cases are very vague. For large data sets and more complex domain models (more rules) we expect even better results.

Comprehensiveness of the approach: It can be shown that calculating the similarity in a hierarchy (e.g. the hierarchy of topics) we used in [5], is equivalent to determining the relevance by applying our approach only on the transitive relation which models that hierarchy (e.g. on the relation `subtopicOf`). Since the transitivity of a relation is modelled in an ontology through the transitivity axiom, the derivation tree which corresponds to the query regarding that relation (e.g. "`CBR` is a `subtopicOf` `KM`") is equal to the path between these two nodes in the hierarchy. This path is used as the key factor for calculating similarity in our previous approach [5]. In other words, our previous approach for calculating similarity can be treated as a special case (when only the transitive axiom exists in the domain ontology) of our new approach (which uses more semantics - all axioms from the domain ontology).

Complexity of the approach: Our approach is general and can be applied to any type of evaluation which relies on an abstract model of the derivation tree (like Definitions 7–11). However, the prerequisite for the approach is the external readability of derivation trees of answers which are returned for a query. Here we discuss only the complexity of the processing the derivation trees, but not the complexity of producing (externalisation) of these structures. As the results presented in the table 1 show, in the case of the Ontobroker system, the externalisation of derivation trees is not a time-consuming activity.

For computing the time-complexity of our approach we make the following assumptions:

- Since the number of instances in an ontology can be very large, they should be stored in a database. It means that the access to the ontology has to be considered as a time-consuming activity.
- The size of the ontology can be approximated using: $\boldsymbol{l}$ (average number of the premises in a rule), $\boldsymbol{k}$ (the average arity of an ontological relation), $\boldsymbol{g}$ (average number of the instantiations of a relation), $\boldsymbol{m}$ (average depth of a derivation tree)
- We consider the case that $\boldsymbol{n}$ results are retrieved for the user's query

The time-complexity for the calculation of the *Ambiguity parameter*, (4), is $\Omega(\boldsymbol{g}*(\boldsymbol{k}\text{-}1))$ – it is the number of accesses to the ontology. The time-complexity for the calculation of the *Specificity parameter*, (5), is $\Omega(\boldsymbol{g}*(\boldsymbol{k}\text{-}1)*\boldsymbol{k})$. The complexity for processing the derivation tree of a result, (8)-(10), is $\Omega(l^m * \boldsymbol{g}*(\boldsymbol{k}\text{-}1)*\boldsymbol{k})$.

Finally, the time-complexity of ranking all $\boldsymbol{n}$ results is $\Omega(\boldsymbol{n}* l^m * \boldsymbol{g}*(\boldsymbol{k}\text{-}1)*\boldsymbol{k})$.

However, the exponential parameter, $\boldsymbol{m}$ - the average depth, is in practice very low (<10), as well the parameter $\boldsymbol{l}$. Further, for our application domain, Semantic Web, the value of the parameter $\boldsymbol{k}$ is 2. The only parameter which can be very large is the number of instances, $\boldsymbol{g}$, i.e. $\boldsymbol{m}$, $\boldsymbol{l}$, $\boldsymbol{k}$ $l^m << \boldsymbol{g}$. Therefore, the time-complexity of our

ranking approach is $\Omega(n*g)$, i.e. it is linear regarding g. It means that our approach scales well with increasing the number of instances in the ontology.

5 Related Work

Semantic methods in Information Retrieval. Using more structured and machine readable information about a web document for improving searching for information is very promising research area in the Semantic Web community. In [15] is presented an approach for information retrieval over documents that consist of both free text and semantic annotations with statements in DAML+OIL. These statements provide both structured and semi-structured information about the documents and their content. The developed framework advocates the interdependency of search (text retrieval/extraction methods) and inference for the precise retrieval over the semantic content. However, the approach does not consider the ranking of retrieved documents.

Ontologies. Although the ideas for ontology-based ranking come from our research in ontology-based systems, it is difficult to compare it in that context, since, as far as we know, none of the inference engines performs ranking of inferred results. Regarding the ranking algorithm itself, we can compare our measure of similarity for the transitive relations (similarity in the hierarchy) with the work done in the NLP community [16] which refers to the similarity between two concepts in a `isA`-taxonomy such as the WordNet or CYC upper ontology. Our approach differs from this notion of similarity in two main aspects: Firstly, our similarity measure is applicable to a hierarchy which may, but need not be a taxonomy and secondly it takes into account not only similarities but also differences between the items being compared, expressing both in semantic-cotopy terms. The second property enables the measuring of self-similarity and subclass-relationship similarity, which are crucial for comparing results derived from the inferencing processes.

A very interesting approach for processing ontology-based information is given in [17]. It exploits the connectionistic structure of an ontology in order to define connectedness metrics between instances in the ontology. The approach is applied for defining communities of practice in the system called ONTOCOPI. The insight behind the approach ONTOCOPI is that if an ontology of the working domain of an organisation is created, then the links between the instances can be measured to indicate which are closely related. However, this approach does not exploit all the semantics of a domain expressed in an ontology (e.g. rules) or which can be induced from the usage of the link structure.

In [18] authors describe an interesting approach for querying Semantic Associations on the Semantic Web, which capture complex relationships between entities that capture a connectivity of entities or a pattern of entities and relationships between them based on a specific notion of an isomorphism called ρ-isomorphism. The approach builds upon a formal model for RDF, which uses a graph data model and also provides a type system for RDF data. Since this model do not encompass the notion of rules, all captured associations are derived from the explicit information existing in a RDF graph. On the other side, the strength of our approach lies in exploiting implicit knowledge, captured in the ontology axioms, for determining

relevance. By materialising all implicit knowledge, lots of information important for determining relevance of facts is lost. Obviously, the differences arise due to different tasks of the approaches: their approach detects complex associations that may be buried deep in the data, whereas our approach is focused on assessing existing associations. Moreover, by allowing querying for relations (e.g. FORALL X <-rst[X->>KM]) our ranking approach could be adapted to this detection task as well.

6 Conclusion

Since the strength of an ontological structure lies in the formal and explicit semantics of the relationships between ontological entities, the Semantic Web has to exploit the full potential of the semantic-based hyperlink structure between web resources, in order to improve the efficiency of resource retrieval. The determination of the relevance of a web resource for a user's query is the issue which can be significantly improved in the Semantic Web, especially the relevance based on the analysis of the inferencing process. In this paper we presented a novel approach for determining relevance in the ontology-based searching for information. This approach is oriented towards the determination of the link relevance and reflects the semantic link-based nature of the Semantic Web. It easily enables the incorporation of any additional information related to a link, such as the usage of that link. Moreover, we see the calculation of the relevance as a task/problem-sensitive decision and we plan to develop task-oriented strategies for calculating relevance.

Acknowledgement. The research presented in this paper would not have been possible without our colleagues and students at the Institute AIFB, University of Karlsruhe. Special thanks to the Jürgen Angele from Ontoprise GmbH for developing the explanation features in the Ontobroker. Research for this paper was partially financed by BMBF in the project "SemIPort" (08C5939) and by EU in the IST-2000-28293 project "Ontologging".

References

[1] Berners-Lee, T.: Business Model for the Semantic Web. W3C, October 2001.
[2] Fensel, D., Bussler, C Y. Ding, and B. Omelayenko: The Web Service Modeling Framework WSMF, Electronic Commerce Research and Applications, 1(2), 2002.
[3] Guarino, N., Masolo, C. and Vetere, G. "OntoSeek: Content-Based Access to the Web", IEEE Intelligent Systems, 14(3), pp. 70–80, (May 1999).
[4] Decker, S., Erdmann, M., Fensel, D. and Studer, R. Ontobroker: Ontology Based Access to Distributed and Semi-Structured Information. In R. Meersman et al., editors, Database Semantics: Semantic Issues in Multimedia Systems, pages 351–369. Kluwer, 1999.
[5] Stojanovic, N., Maedche, A., Staab, S., Studer, R. and Sure, Y. "SEAL — A Framework for Developing SEmantic PortALs", ACM K-CAP 2001. October, Vancouver, 2001.

[6] Bozsak, E., Ehrig, M., Handschuh, S., Hotho, A., Mädche, A., Motik, B., Oberle, D., Schmitz, C., Staab, S., Stojanovic, L., Stojanovic, N., Studer, R., Stumme, G., Sure, Y., Tane, J., Volz, R. and Zachariasm V. KAON – Towards a large scale Semantic Web. In: Proceedings of EC-Web 2002. Aix-en-Provence, France, September 2–6, 2002. LNCS, Springer, 2002.

[7] Stojanovic, N., Stojanovic, L., Gonzalez, J. On Enhancing Searching for Information in an Information Portal by Tracking Users' Activities, First International Workshop on Mining for Enhanced Web Search (MEWS 2002), held in conjunction with 3rd International Conference on Web Information System Engineering WISE 2002, Singapore, 2002, IEEE Press.

[8] Boley, H., Tabet, S., Wagner, G. Design Rationale of RuleML: A Markup Language for Semantic Web Rules, SWWS'01, Stanford University, July/August 2001.

[9] OWL Web Ontology Language 1.0 Reference, W3C Working Draft 29 July 2002.

[10] Kifer, M., Lausen, G., Wu, J.: Logical Foundations of Object-Oriented and Frame-Based Languages, Journal ACM, 42:741–843, 1995.

[11] Kulyukin, V., Settle, A Ranked retrieval with semantic networks and vector spaces. JASIST 52(13): 1224–1233 (2001).

[12] Ullman J. D. Principles of Database and Knowledge-based Systems. Computer Science Press, Rockville.

[13] Martin, C. Direct memory access parsing (Tech. Rep. No. CS93-07). Chicago, IL: The University of Chicago, Department of Computer Science.

[14] Joachims, T. Evaluating Retrieval Performance Using Clickthrough Data, Proceedings of the SIGIR Workshop on Mathematical/Formal Methods in Information Retrieval, 2002.

[15] Shah, U., Finin, T., Joshi, A., Cost S. and Mayfield, J. Information Retrieval on the Semantic Web, ACM CIKM'02, 2002.

[16] Resnik, P. Using information content to evaluate semantic similarity in a taxonomy. In Proceedings of IJCAI-95, Montreal, Canada, 1995.

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

[18] Anyanwu K. and Sheth A. r-Queries: Enabling Querying for Semantic Associations on the Semantic Web, The Twelfth International World Wide Web Conference, Budapest, Hungary, pp. 690–699, May 2003