Information Processing and Management 50 (2014) 132–155

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

Effective ranking and search techniques for Web resources considering semantic relationships ☆

Jihyun Lee [a], Jun-Ki Min [b], Alice Oh [a], Chin-Wan Chung [a,*]

[a] Korea Advanced Institute of Science and Technology, Yuseong-gu, Guseong-dong, Daejeon 305-701, Republic of Korea
[b] Korea University of Technology and Education, Byeongcheon-myeon, Chungnam 330-708, Republic of Korea

A R T I C L E I N F O

Article history:
Received 3 July 2010
Received in revised form 27 August 2013
Accepted 28 August 2013

Keywords:
Semantic search
Ranking
Semantic relationship
Ontology
Semantic Web

A B S T R A C T

On the Semantic Web, the types of resources and the semantic relationships between resources are defined in an ontology. By using that information, the accuracy of information retrieval can be improved.

In this paper, we present effective ranking and search techniques considering the semantic relationships in an ontology. Our technique retrieves *top-k* resources which are the most relevant to query keywords through the semantic relationships. To do this, we propose a weighting measure for the semantic relationship. Based on this measure, we propose a novel ranking method which considers the number of meaningful semantic relationships between a resource and keywords as well as the coverage and discriminating power of keywords. In order to improve the efficiency of the search, we prune the unnecessary search space using the length and weight thresholds of the semantic relationship path. In addition, we exploit Threshold Algorithm based on an extended inverted index to answer top-*k* results efficiently. The experimental results using real data sets demonstrate that our retrieval method using the semantic information generates accurate results efficiently compared to the traditional methods.

1. Introduction

With the massive growth of the Web, we have been confronted with a flood of information, and hence search engines have become one of the most helpful tools for obtaining desired information from the Web. The keyword-based search method has been the most popular search method in the search engines since it provides simple and user friendly interface. The keyword-based search method requires users to input several keywords describing the search target and returns the search results containing the keywords.

In general, the keyword-based search determines the relevance of resources for query keywords mainly based on the occurrence of the keywords in their textual descriptions (e.g., title, body, anchor text, and so on). It cannot ensure that the returned results preserve the semantic relationships among the keywords which users have intended when submitting the keywords (Li et al., 2007). Because of this reason, current search engines sometimes miss highly relevant results and return some irrelevant results for user requests. For example, consider a query looking for laboratories in Europe researching on Semantic Web. You may input the following keywords: 'Laboratory', 'Semantic Web', and 'Europe'. For this query, one of the most popular search engines returns about 2,360,000 pages. However, on the top 20 pages, only four pages are related to

☆ A short preliminary version of this paper was published in the proceeding of WWW 2009 as a two page poster paper.
* Corresponding author. Tel.: +82 42 350 3537; fax: +82 42 350 7737.
E-mail addresses: jihyun.s.lee@gmail.com (J. Lee), jkmin@kut.ac.kr (J.-K. Min), alice.oh@cs.kaist.ac.kr (A. Oh), chungcw@kaist.edu (C.-W. Chung).

0306-4573/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.ipm.2013.08.007

the desired laboratories and other irrelevant pages just contain some of the three keywords, such as 'W3C Semantic Web FAQ' and 'Semantic Web Europe'. In the irrelevant pages, the relationship *'research on'* between 'Laboratory' and 'Semantic Web' and the relationship *'located on'* between 'Laboratory' and 'Europe' are not preserved. Furthermore, the diverse implicit meanings in the relationships among keywords are ignored in the search. For example, *'research on'* implies some indirect relationships via 'published papers' about Semantic Web and 'researchers' studying on Semantic Web. As a result, many relevant pages have been pushed down in the ranked list.

We propose a semantic search framework to overcome such limitations of the traditional keyword-based search by enriching the search process with an ontology, which is one of the purposes of the Semantic Web. An ontology is a formal knowledge description of concepts and their relationships. The semantic relationships between resources and keywords could be extracted by traversing the ontology. The extracted semantic relationships can complement the keyword-based search method. Our semantic search framework extends keyword-based search by using ontologies, with the aim of finding resources relevant to query keywords through the semantic relationships. The semantic search makes hidden relationships between the words of desired resources and keywords explicit by using diverse semantic relationships defined in the ontology, thereby it can effectively access the relevant resources and rank them. Consequently, we expect that the precision and recall of the search would be improved. Recently, Google has started to support a primitive semantic search based on the knowledge graph which is similar to the ontology. Google enhances its search service by augmenting the search results with sets of associated facts based on the knowledge graph. The usefulness and feasibility of our semantic search could be confirmed by this attempt.

This semantic search would be useful when a sufficient ontology associated with the search domain is prepared in advance. As more resources and their relationships in the domain become well defined, more comprehensive semantic search would become feasible. In addition, in order to effectively limit the search scope, we assume that each query contains the type (i.e., a class in an ontology) of the desired resource such as *Publication* or *Professor*.

Fig. 1 shows the overall search process in our framework. Before searching documents (e.g., biography or web page), the ontology generator constructs an ontology describing the contents in the collected documents. For this work, we assume that the ontology has already been constructed. Note that there has been active research on the ontology construction (Kiryakov et al., 2003; Ceravolo and Damiani, 2007; Suchanek et al., 2007). Given a user query containing a type **T** and a set of keywords $\{k_1, k_2, \ldots, k_n\}$, the semantic search engine finds the relevant resources through the exploration of the ontology and returns a ranked list of the URIs of the resources in the order of their relevance. Finally, the document retriever retrieves the documents corresponding to the returned URIs.

Since we consider the semantic relationships between the resources and the query keywords in the search process, the ranking of the set of results should reflect how well each semantic relationship discriminates a result from the other results in the set. Thus, we devise a novel weighting measure for the semantic relationships, such that it assigns a higher weight to semantic relationships with less ambiguity in identifying the target resources. Further, we design a novel ranking model for resources by considering the following three major relevance criteria: the number of important relationships between resources and query keywords, the coverage of the keywords, and the discriminating power of the keywords. If there are many kinds of semantic relationships from resources to query keywords in an ontology, it would take much time to examine the entire set of relationships in the semantic search. In order to improve the efficiency of our semantic search, we prune the search space using the length and importance of the semantic relationship. Our semantic search may generate many results that are related to the query keywords. Therefore, we adapt the Threshold Algorithm (Fagin et al., 2003) to efficiently retrieve the *top-k* results without examination of the entire result set.

The main contributions of our work are summarized below:

- *A weighting method for semantic relationships:* We propose a weighting measure for semantic relationships which assigns a higher weight to significant relationships having a higher level of contribution to discriminate the answer. In our work, the weight is automatically computed without the intervention of domain experts. Thus, the weighting method could be applied to large and complex ontologies.

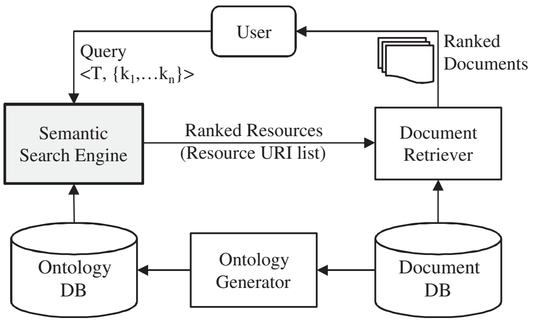


Fig. 1. Document retrieval process based on ontologies.

- *A novel ranking method:* We suggest a novel ranking method considering the number of meaningful semantic relationships, the coverage of the keywords, and the discriminating power of the keywords. These are fundamental and important criteria for users to judge the relevance of a search result. It is the first time to consider all three criteria for ranking results in the semantic search. The accuracy of our ranking method is proven by the experiments comparing with the existing ranking methods based on an ontology.
- *Efficient search techniques:* For efficiency of the semantic search engine, we prune meaningless relationships by using the length and weight of the semantic relationship. Our pruning technique reduces the search space by average 40–45% in the experiments. In addition, we construct an extended inverted index, 'keyword index', in advance, to avoid the expensive traversal of a large ontology instance graph during the answering process. Furthermore, we enhance the efficiency of the *top-k* answering by using Threshold Algorithm based on the keyword index.

The rest of the paper is organized as follows: Section 2 discusses the related work. Section 3 explains the data model for an ontology and defines the semantic search. Section 4 describes the semantic search framework. Section 5 introduces a weighting method for semantic relationships and a new ranking method for the semantic search. Section 6 presents a pruning method to improve the accuracy and efficiency of the semantic search and a *top-k* query processing method. Section 7 shows the empirical evaluation of our search method. Finally, in Section 8, we conclude our work.

2. Related work

On the Semantic Web, the semantic relationships among resources can be various and complex. Anyanwu and Sheth (2003) formalized the various complex relationships between resources. Anyanwu et al. (2005) proposed a ranking method, called SemRank, to rank semantic relationships between two resources based on the predictability of the relationships. These methods focus on the retrieval and ranking of relationships between a given single pair of resources.

Recently, some techniques considering the semantic relationships among resources expressed in an ontology in order to improve the accuracy of the keyword-based search have been conducted. Li et al. (2007) addressed the problem of the current keyword-based search (i.e., keywords-isolated pages which only include the query keywords but have no relationship with the query in the context) due to the ignorance of relationships among keywords. They introduced a relation-based search engine, OntoLook. OntoLook constructs a concept-relation graph consisting of concepts (i.e., classes) of keywords and all semantic relationships among them. Then, OntoLook finds matches of the graph from the ontology database and returns URIs and values corresponding to the nodes in the graph as an answer. OntoLook reduces many keywords-isolated pages by considering relationships among keywords. However, OntoLook demands a tedious work to input a query since users should input concepts for all keywords. In addition, in order to reduce search space, OntoLook cuts some arcs (i.e., relations) from the concept-relation graph, but the algorithm does not consider the semantics and importance of the arcs. Besides, OntoLook does not provide any ranking method to rank retrieved results.

Castells et al. (2007) presented an information retrieval framework using an ontology in order to improve the accuracy. Basically, documents are annotated and an ontology is constructed based on the annotations. Then, the annotations for each document are weighted by an adaptation of the *tf*idf* measure. For a user query, the annotations matched to the query are retrieved from the ontology, and then the documents containing the annotations are selected as the query results. The result documents are ranked by an adapted vector-space model which assigns higher scores to the documents containing many high weighted annotations. In this work, the relationship between the annotations was used to find the relevant document through the ontology query processing, but the differences among the weights of the relationships were not considered.

In the field of the ontology query processing, Stojanovic et al. (2003) proposed a ranking method for the results of an ontology query. Rocha et al. (2004) proposed a spread activation algorithm which finds additional relevant results by using an ontology after a traditional keyword-based search. These methods determine the relevance based on a link analysis where the amount and specificity of the relationships are considered, but they are insufficient for precise ranking. For example, *'be interested in'* and *'write a publication about'*, which are the semantic relationships between a researcher and a research topic, have different weights to explain a person. *'write a publication about'* is more specific and more informative descriptor to explain a researcher than *'interested in'*. We define the semantic diversity of relationships as the semantically different importance of the relationships from a resource to a keyword to determine the relevance of the resource against the keyword. The different weights of relationships according to the semantic diversity should be reflected to determine the relevance of a resource. However, the ranking method in Stojanovic et al. (2003) did not consider the diversity of semantic relationships. The work in Rocha et al. (2004) assigned a weight to each relationship (i.e., property) according to its semantics. However, the weight is determined by domain experts and it is impractical to manually assign the weights to all relationships in case that the ontology is large and complex. Besides, queries consisting of multiple keywords with different importance to determine the relevance are not effectively handled.

The keyword-based search in various data models has also been extensively studied in the literature. Representatively, Guo et al. (2003); Hristidis and Papakonstantinou (2002); Liu et al. (2006); Luo et al. (2007); Zhou et al. (2012); and Theobald et al. (2005) proposed ranking methods considering the link structure covering query keywords. The link structure is composed of primary-foreign key relationships in the relational model, edges in the graph model, and parent–child relationships in the XML model. However, these methods also do not consider the various features of links such as the specificity and

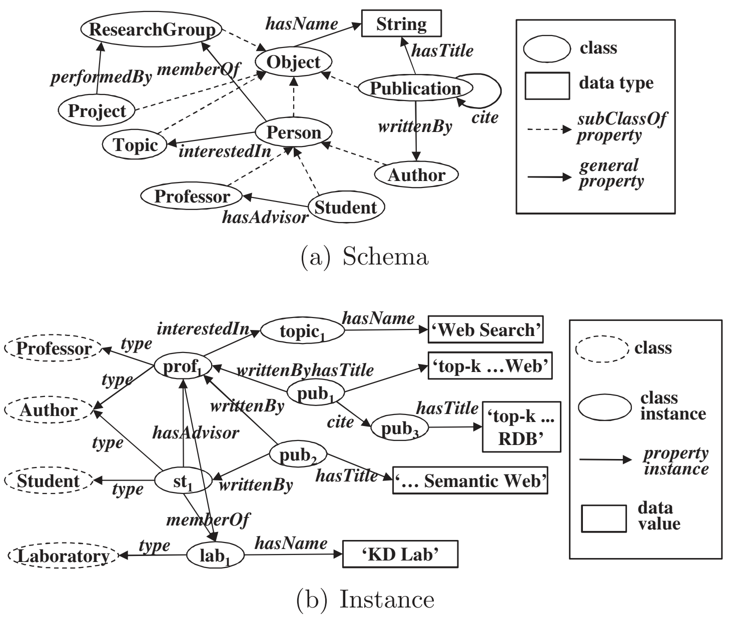


Fig. 2. An example of an ontology.

semantic diversity. Guo et al. (2003) and Theobald et al. (2005) considered the specificity of elements or search terms in their relevance models instead of the links.

In order to determine the weight of a semantic relationship, the importance of the components composing the semantic relationship must be determined. Alani et al. (2006) proposed weighting measures for a class in various aspects such as the number of semantic neighborhoods and centrality of the class. However, this study did not deal with the weighting of a property. Wu et al. (2008) presented a ranking method for classes and properties in an ontology. This study introduced the features of important classes and properties and designs a ranking model based on the features. The ranking model has the reverse mechanism of PageRank, and the weights of classes and properties reinforce each other in the ranking model in an iterative manner. This study only focused on the effective identification of potentially important classes and properties in an ontology in order to facilitate user's interpretation of the ontology. Therefore, the ranking model is insufficient to be directly applied to the weighting measure for semantic relationships.

3. Preliminaries

3.1. Data model

An ontology is composed of a schema and its instance to represent knowledge description of concepts and their relationships. The schema defines classes (i.e., concepts) and properties which are the relationships between classes. In the instances of the schema, class instances[1] and property instances are declared according to the schema.

For example, Fig. 2 shows a part of the ontology describing resources and their relationships in a research domain. The schema in Fig. 2(a) defines classes existed in the research domain such as *Publication* and *Professor* as well as properties such as *writtenBy*. In an instance of the schema in Fig. 2(b), the resource $prof_1$ is an instance of the *Professor* class, the resource pub_1 is an instance of the *Publication* class, and $pub_1 - writtenBy - prof_1$ is an instance of the *writtenBy* property. In addition, we say that there is a semantic relationship between two resources if they are connected via properties.

In this study, we consider an ontology represented in a subset of OWL-Lite which includes RDF features, object/datatype property, and inverseOf property. These features are sufficient to capture the semantic relationships among resources needed in our method. Since we do not consider more powerful ontology languages like OWL-DL, our semantic search method cannot extract and use hidden semantic relationships which would only be revealed by inferences. On the other hand, since our semantic search method requires only the basic features of the ontology language, it has fewer restrictions for generality and deployment. In addition, we assume that all classes, properties, data types, and instances are explicitly identified in the ontology, and a property has one domain and one range. The formal definitions for the ontology are as follows:

[1] We use 'class instance' and 'resource' interchangeably since the search target is a resource which is denoted as a class instance in an ontology.

Definition 1 (*Schema*). Schema S is defined as $\langle C, D, P\rangle$. C is the set of classes, D is the set of data types, and P is the set of properties. All classes, properties, and data types are explicitly identified by URIs. For $d \in C$, $r \in C \cup D$, there can be a property $p(d,r) \in P$. For each property $p(d,r) \in P$, d is the *domain* and r is the *range* of p.

There are two types of properties: the *object property* which has a class as its range and the *data property* which has a data type like a string or a numeric value as its range. In our work, we assume that the range of a data property is string for simplicity. The hierarchies among classes and among properties are defined by the *subClassOf* property and the *subPropertyOf* property, respectively. In addition, we assume that each property $p(d,r)$ in the ontology has its own inverse property of $p^{-1}(r,d)$.

Definition 2 (*Instance Graph*). An instance graph confined to a schema $S = \langle C, D, P\rangle$ is defined as a directed graph $G = \langle V, E\rangle$. V is the set of instances. $[c]$ indicates the set of instances of $c \in C \cup D$. A resource denotes a class instance in the instance graph. For each $v \in V$, $v \in [c]$ if $v.type = c$. E is the set of relationships among instances in V. $[p(d,r)]$ denotes the set of property instances of $p(d,r) \in P$. For each $e(v_i, v_j) \in E$, $e(v_i, v_j) \in [p(d,r)]$ if $e = p$, $v_i \in [d]$ and $v_j \in [r]$. v_i is the subject and v_j is the object of e. All resources are explicitly identified by URIs.

The set of instances of a class c includes the instances of its sub-classes in the class hierarchy, and the set of instances of a property p includes the set of instances of its sub-properties in the property hierarchy.

Definition 3 (*Semantic Path*). A semantic path sp is a sequence of properties $p_1(d_1,r_1), \ldots, p_m(d_m,r_m)$ in a schema $S = \langle C, D, P\rangle$, where $p_i(d_i,r_i) \in P$ and r_i and d_{i+1} are the same class or have a common super class (excluding the root) in the class hierarchy.

An instance graph G confined to schema S can include matches of a semantic path in S. A match of the semantic path is a sequence of property instances, which is called *semantic path instance*.

Definition 4 (*Semantic Path Instance*). For a semantic path $sp = p_1(d_1,r_1), \ldots, p_m(d_m,r_m)$, $ip = e_1(s_1,o_1)\ e_2(s_2,o_2), \ldots, e_m(s_m,o_m)$ is a semantic path instance of sp if $e_i(s_i,o_i) \in [p_i(d_i,r_i)]$ and $o_i = s_{i+1}$ for all e_i. s_1 is the source of ip and o_m is the destination of ip. $[sp]$ denotes the set of all semantic path instances of sp.

In Fig. 2, $writtenBy^{-1}(Author, Publication)hasTitle(Publication, String)$ s a semantic path, and $writtenBy^{-1}(prof_1, pub_1)hasTitle(pub_1, \text{'}top - k \ldots Web\text{'})$ is a semantic path instance[2] of the semantic path.

In an instance graph, the existence of a semantic path instance from v_i to v_j implies that v_i can be described by v_j through the semantic path instance.

3.2. Semantic search

In order to complement the traditional keyword-based search, the semantic search considers the type of the desired resource and the semantic relationships among resources.

In the semantic search, a user query Q consists of (1) a class T of resources in which a user is interested and (2) a set of keywords K which describes the desired resources. Thus, the answer of Q is a set of resources whose class is T and which are related to the set of keywords K through the semantic path instances.

In fact, data values are leaf nodes in an instance graph and may contain some query keywords. Since the answer of the semantic search is the set of resources connected to data values containing query keywords, the data values are not the target of the semantic search.

Definition 5 (*Semantic Search*). Given a schema $S = \langle C, D, P\rangle$ and an instance graph G confined to S, the semantic search is to find the answer A for query $Q = \langle T, K\rangle$, where $T \in C$. For each resource $a \in A$, there should be at least one semantic path instance from resource a to data value s in G where $a \in [T]$ and value s contains $k \in K$. $IP(a,k)$ denotes the set of such semantic path instances from a to s including any query keyword k.

If we do not care about the resource type in the semantic search, we use the root class (i.e., owl:Thing) of the class hierarchy as the desired type T. It means that all resources in the instance graph belong to the search scope.

4. Semantic search framework

In this section, we briefly explain the semantic search framework. Fig. 3 shows the main components of the semantic search engine and the corresponding components of the traditional IR. The explanations for the components are as follows:

- *Input:* It is a query (T, K), where T is the desired resource type and K is the set of keywords.

[2] We ignore the semantic path instance containing a cycle like '$hasAdvisor(prof_1, st_1)writtenBy^{-1}(st_1, pub_2)writtenBy(pub_2, prof_1), \ldots$' in our work since such semantic path instances are redundant.

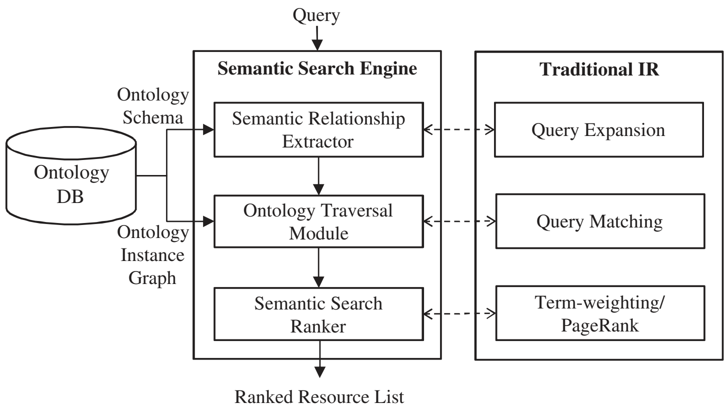


Fig. 3. The major components of the semantic search engine.

- *Semantic Relationship Extractor:* This module finds the possible semantic relationships between the target resources and the query keywords. It extracts the set of semantic paths *SP* from *T* to *K* from the ontology schema. It expands a basic keyword query to a form of semantically enhanced query (i.e., a set of semantic relationships) based on the ontology schema. Thus, this module corresponds to the query expansion component in the traditional IR system. The details of the extraction process and a strategy to prune unnecessary semantic paths are presented in Section 6.1.
- *Ontology Traversal Module:* This module retrieves the resources that match the user query by traversing an ontology instance graph along the extracted semantic relationships. It finds the set of resources *R* each of which reaches some keywords in *K* through *SP* from an instance graph. This module corresponds to the query matching component in the traditional IR system.
- *Semantic Search Ranker:* This module is in charge of ranking resources in the order of their relevance. For each resource $r_i \in R$, it computes $Rank(r_i,K)$ which is the relevance of resource r_i for *K*. This module corresponds to the ranking component using Term-weighting (i.e., *tf*idf*) or *PageRank* in the traditional IR system. The details of the ranking method as well as the relevance criteria for the semantic search are presented in Section 5
- *Output:* It is a list of ranked resources in descending order of $Rank(r_i,K)$. The efficient method retrieving *top-k* results with the highest relevance scores is explained in Section 6.2.

5. Semantic search ranking

The number of relevant results returned by the semantic search is usually very big. In general, however, users are interested in only the first *k* results listed. Therefore, the semantic search should provide the results in the order of relevance. Which results are more relevant to the user query? Intuitively, the relevance of a candidate result can be judged based on the semantic relationships between the result and the query keywords. To determine the relevance of a result, we consider the following criteria:

- *The number of meaningful semantic paths:* A resource, which has many semantic paths directed to query keywords, is more relevant. In addition, the importance of the semantic paths should also be considered.
- *The coverage of keywords:* Users describe what they want to find by using query keywords. Therefore, a resource which reaches to as many query keywords as possible through semantic paths is more relevant.
- *The discriminating power of keywords:* A resource having semantic paths to query keywords, which distinguish relevant resources from others, is more relevant.

By using these relevance criteria, we design a new ranking method for the semantic search. In the following sections, we will present the measures for these criteria and explain how to reflect the criteria in our method.

5.1. Weighting method for semantic paths

The semantic path which can effectively distinguish the relevant resources from others is a meaningful semantic path for the semantic search. According to this, we propose a weighting measure for semantic paths.

5.1.1. The weight of a property

A semantic path consists of one or more properties. Therefore, in order to measure the weight of a semantic path, we should be able to determine the weight of each property in the semantic path. The weight of a property represents how well the property describes its subject. The weight of a property is determined according to (1) the ability of the property to discriminate the candidate set of its subject from others and (2) the identification power of the property about the subject when we know the object, and vice versa. Since each factor is not sufficient alone to determine the weight of a property, we combine these two factors to complement each other.

In the ontology of Fig. 2, if a property is *hasName*, we can expect that its subject is an *Object*. In case of *hasTitle*, we can reduce the candidate set of the subject to *Publication*. Thus, *hasTitle* is more discriminating property. We consider this discriminating power of a property to determine the weight of the property.

In information theory, the information content gained by the occurrence of an event x can be quantified as:

$$I(x) = -\log_2 pr(x)$$

where $pr(x)$ is the occurring probability of x (Resnik, 1999). This represents the uncertainty eliminated by the occurrence of the event x. As x occurs rarely, x is more informative. In our domain, we adapt this measure to determine the discriminating power of a property.

Based on information theory, the amount of information gained by the existence of a property is calculated as follows:

For an instance graph $G = \langle V,E\rangle$ confined to the schema $S = \langle C,D,P\rangle$, the probability $pr(p(d,r))$ that a resource is a subject of a property $p(d,r) \in P$ is computed as follows:

$$pr(p(d,r)) = \frac{|sub(p(d,r))|}{|N|}$$

where $sub(p(d,r)) = \{v_i—e(v_i, v_j) \in [p(d,r)]\}$ and N is the set of resources excluding data values.

Thus, the amount of information contained in a property $p(d,r)$ is

$$I(p(d,r)) = -\log_2 pr(p(d,r)) \tag{1}$$

For example, consider an ontology instance complying with the schema in Fig. 2(a). We assume that there are 1000 resources including people, publications, and topics in the ontology, 600 people are interested in some topics, and 100 people (i.e., authors) wrote some publications. The amount of information contained in *interestedIn* and *writtenBy*$^{-1}$ is obtained as follows:

$$I(interestedIn(Person, Topic)) = -\log_2 pr(interestedIn(Person, Topic)) = -\log_2 \frac{600}{1000} = -\log_2 0.6 \approx 0.73.$$
$$I(writtenBy^{-1}(Author, Publication)) = -\log_2 pr(writtenBy^{-1}(Author, Publication)) = -\log_2 \frac{100}{1000} = -\log_2 0.1 \approx 3.32.$$

This amount of information contained in a property represents the discriminating power of the property.

The discriminating power of a property is not sufficient to determine the importance of the property. For example, it tends to underestimate the actual importance of *hasName* compared with other properties of *Person* such as *memberOf* since every person has its own name while only some people are members of a research group. However, *hasName* can identify a person by a given name, but *memberOf* cannot identify a person by a given research group. We consider this identification power of a property to determine the weight of the property.

In information theory, the amount of information that one random variable contains about another random variable is measured by Mutual Information (Cover, 1991). To measure the identification power of a property about its object given a subject, and vice versa, we adapt the mutual information measure.

The mutual information between the domain d and the range r for a property $p(d,r)$ is

$$MI(p(d,r)) = \sum_{o\in r}\sum_{s\in d} pr(s,o)\cdot \log_2\left(\frac{pr(s,o)}{pr(s)pr(o)}\right)$$

where the sample space is $[p(d,r)]$, $pr(s)$ is the probability that $e \in [p(d,r)]$ has s as its subject and $pr(o)$ is the probability that $e \in [p(d,r)]$ has o as its object. In addition, $pr(s,o)$ is the probability that $e \in [p(d,r)]$ has s and o as its subject and object respectively at the same time. Thus, for $p(d,r)$, $pr(s) = \frac{|\{e(v_i, v_j)|v_i = s \wedge e(v_i, v_j) \in [p(d,r)]\}|}{|[p(d,r)]|}$. $pr(o)$ and $pr(s,o)$ are obtained by the similar way.

For example, the mutual information between *Author* and *Publication* for the property *writtenBy*$^{-1}$ and the mutual information between the *Person* and *String* for the property *hasName* in Fig. 4 can be obtained as follows:

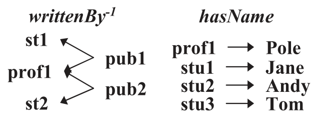


Fig. 4. An example of instances of properties *writtenBy*$^{-1}$ and *hasName*.

$$\begin{aligned}MI(writtenBy^{-1}(Author, Publication)) &= pr(st_1, pub_1)\cdot\log_2\left(\frac{pr(st_1, pub_1)}{pr(st_1)pr(pub_1)}\right) + pr(prof_1, pub_1)\\ &\cdot\log_2\left(\frac{pr(prof_1, pub_1)}{pr(prof_1)pr(pub_1)}\right) + pr(prof_1, pub_2)\cdot\log_2\left(\frac{pr(prof_1, pub_2)}{pr(prof_1)pr(pub_2)}\right)\\ &+ pr(st2, pub2)\cdot\log_2\left(\frac{pr(st2, pub2)}{pr(st2)pr(pub2)}\right)\\ &= \frac{1}{4}\cdot\log_2\left(\frac{\frac{1}{4}}{\frac{1}{4}\cdot\frac{1}{2}}\right) + \frac{1}{4}\cdot\log_2\left(\frac{\frac{1}{4}}{\frac{1}{2}\cdot\frac{1}{2}}\right) + \frac{1}{4}\cdot\log_2\left(\frac{\frac{1}{4}}{\frac{1}{2}\cdot\frac{1}{2}}\right) + \frac{1}{4}\cdot\log_2\left(\frac{\frac{1}{4}}{\frac{1}{4}\cdot\frac{1}{2}}\right) = 0.5.\end{aligned}$$

$$\begin{aligned}MI(hasName(Person, String)) &= pr(prof_1, Pole)\cdot\log_2\left(\frac{pr(prof_1, Pole)}{pr(prof_1)pr(Pole)}\right) + pr(st_1, Jane)\cdot\log_2\left(\frac{pr(st_1, Jane)}{pr(st_1)pr(Jane)}\right)\\ &+ pr(st_2, Andy)\cdot\log_2\left(\frac{pr(st_2, Andy)}{pr(st_2)pr(Andy)}\right) + pr(st_3, Tom)\cdot\log_2\left(\frac{pr(st_3, Tom)}{pr(st_3)pr(Tom)}\right)\\ &= \frac{1}{4}\cdot\log_2\left(\frac{\frac{1}{4}}{\frac{1}{4}\cdot\frac{1}{4}}\right) + \frac{1}{4}\cdot\log_2\left(\frac{\frac{1}{4}}{\frac{1}{4}\cdot\frac{1}{4}}\right) + \frac{1}{4}\cdot\log_2\left(\frac{\frac{1}{4}}{\frac{1}{4}\cdot\frac{1}{4}}\right) + \frac{1}{4}\cdot\log_2\left(\frac{\frac{1}{4}}{\frac{1}{4}\cdot\frac{1}{4}}\right) = 2.\end{aligned}$$

In case that there are many property instances, no doubt that the cost of computing the mutual information exactly will be prohibitively expensive. Therefore, we compute the mutual information approximately.

The probability of picking an instance of property $p(d,r)$ from $[p(d,r)]$ is $\frac{1}{|[p(d,r)]|}$. $pr(s)$ is approximately $\frac{1}{|sub(p(d,r))|}$ where $sub(p(d,r))$ is the set of distinct subjects in $[p(d,r)]$ with the assumption that every subject has the same number of objects. $pr(o)$ is also obtained in a similar manner. Thus, the mutual information can be estimated as:

$$MI(p(d,r)) \approx \log_2\frac{\frac{1}{|[p(d,r)]|}}{\frac{1}{|sub(p(d,r))|}\cdot\frac{1}{|obj(p(d,r))|}} \tag{2}$$

where $sub(p(d,r)) = \{v_i — e(v_i, v_j) \in [p(d,r)]\}$ and $obj(p(d,r)) = \{v_j — e(v_i, v_j) \in [p(d,r)]\}$.

For example, the approximated $MI(writtenBy^{-1})$ is $log_2\frac{\frac{1}{4}}{\frac{1}{3}\frac{1}{2}} = 0.58$.

The mutual information tends to be favorable to the property which has a large number of distinct subjects or objects. The combination with the discriminating power of a property compensates this bias.

As mentioned before, each of them alone is not sufficient to determine the weight of a property, and they are complementary each other. In order to combine I and MI, we can consider various aggregation methods such as average, summation, and multiplication. We choose a weighted summation as a combining method for adjusting the influence of each factor on the overall weight.

By using Eq. (1) and (2), we can compute the weight of a property $p(d,r)$ as follows:

$$w(p(d,r)) = \alpha\cdot I(p(d,r)) + \beta\cdot MI(p(d,r)) \tag{3}$$

where $0 \leqslant \alpha,\ \beta \leqslant 1$ and we normalize I and MI to be in the range $[0,1]$ by $\frac{I(p) - min_{p\in P}I(p)}{max_{p\in P}I(p) - min_{p\in P}I(p)}$ and $\frac{MI(p) - min_{p\in P}MI(p)}{max_{p\in P}MI(p) - min_{p\in P}MI(p)}$, respectively, where P is a set of properties.

5.1.2. The weight of the semantic path

A semantic path is composed of one or more properties. We obtain the weight of the semantic path by combining the weights of constituent properties. For simplicity, we assume that each property is independent of others. As the length of a semantic path gets longer, the relevance between the source and the destination decreases. Therefore, the addition and average functions are not suitable to combine the weights of properties. In order to reflect the path length to the weight, we normalize the property weight to be in $\left[\frac{min_{p\in P}w(p)}{max_{p\in P}w(p)}, 1\right]$ where P is a set of properties. Then, we multiply the normalized weights. From here on, $w(p(d,r))$ is the normalized weight. This combining function (i.e., normalization and multiplication) makes the weight of a semantic path to be in the range $(0,1]$. In addition, we add an attenuation parameter to control the effect of the path length. The weight of semantic path sp is computed as follows:

$$W(sp) = \left(\prod_{p(d,r)\in sp} w(p(d,r))\right)\cdot\delta^{length(sp)-1} \tag{4}$$

where $length(sp)$ indicates the number of properties in sp, and δ is an attenuation parameter which is tunable from 0 to 1.

A semantic path instance has the weight of its corresponding semantic path.

$$W(ip) = W(sp) \quad if\ ip\ \in\ [sp]$$

Consequently, in contrast to previous methods, the weight of a semantic relationship can be computed without the intervention of a domain expert in our work.

5.2. Ranking formula for semantic search

As mentioned in Section 3.2, the relevance of a semantic search result is determined by three important criteria: the number of meaningful semantic path instances, the coverage of query keywords, and the discriminating power of query keywords. In this section, we propose a novel ranking method considering these three criteria.

5.2.1. The number of meaningful semantic path instances

We regard the resources related to query keywords through more meaningful semantic path instances as more relevant results for a query. The meaningfulness of a semantic path instance is represented by the weight of the semantic path obtained by Eq. (4).

The relevance of a resource a for each keyword in K according to the number of meaningful semantic path instances can be computed as follows:

$$R(a,k_i) = \sum_{ip \in IP(a,k_i)} W(ip), \quad \textit{for } k_i \in K \tag{5}$$

Recall that $IP(a,k_i)$ in Definition 5 is the set of all semantic path instances from a to value s including keyword k_i.

However, the relevance measure presented in Eq. (5) has a problem. According to Section 5.1, as the number of property instances increases, the weight of the property is likely to be reduced. Therefore, a semantic path with a small weight tends to have a large number of path instances. This feature can cause a problem such that the relevance score of an irrelevant resource can be greater than that of a relevant one.

Consider, the following two semantic paths and the corresponding weights:

sp_1 : *hasTitle (Publication, String)*
sp_2 : *writtenBy (Publication, Author) writtenBy*$^{-1}$*(Author, Publication) hasTitle (Publication, String)*

$$W(sp_1) = 1 \text{ and } W(sp_2) = 0.15$$

In many cases, the number of semantic path instances of sp_1 is one because each publication usually has a single title. In contrast, the number of instances of sp_2 increases as the number of authors and the number of publications written by the authors increase. For example, the publication pub_1 has the title including the keyword *'xml'*. On the other hand, the publication pub_2 is written by two authors, each author has written 15 publications, and among the publications, 20 publications contain *'xml'* in their titles. However, the title of pub_2 does not contain *'xml'* (i.e., the topic of pub_2 is not *'xml'*). Thus, pub_1 is more relevant to the keyword *'xml'* than pub_2. However, $R(pub_1,\text{'}xml\text{'})$ is 1 and $R(pub_2,\text{'}xml\text{'})$ is 3 (=0.15·20) since the number of semantic path instances of sp_1 for pub_1 is 1 and that of sp_2 for pub_2 is 20.

In order to attenuate this side effect, we reflect the specificity of a semantic path instance. The specificity of a semantic path instance is defined as follows:

Definition 6 (*Specificity of a semantic path instance*). Given a semantic path instance $ip = e_1(s_1,o_1), \ldots, e_m(s_m,o_m) \in [p_1(d_1,r_1), \ldots, p_m(d_m,r_m)]$, the specificity of ip, $spec(ip) = \prod \frac{1}{degree(s_i,p_i)}$, where $degree(s_i,p_i)$ represents the number of instances of a property p_i whose subject is s_i.

In the above example, the specificity of a semantic path instance of sp_2 starting from pub_2 is $\frac{1}{30}\left(=\frac{1}{2}\cdot\frac{1}{15}\cdot 1\right)$. However, the specificity of a semantic path instance of sp_1 starting from pub_1 is 1 since $degree(pub_1, hasTitle) = 1$.

Considering the specificity of a semantic path instance, we modify the relevance measure in Eq. (5) as follows:

$$R(a,k_i) = \sum_{ip \in IP(a,k_i)} (W(ip) \cdot spec(ip)), \quad \textit{for } k_i \in K \tag{6}$$

As applying the specificity of a path instance, the maximum relevance score through a particular semantic path is the weight of the semantic path. For the previous example, $R(pub_2,\text{'}xml\text{'})$ is $0.1\left(= 20 \cdot \left(0.15 \cdot \frac{1}{30}\right)\right)$ and $R(pub_1,\text{'}xml\text{'})$ is 1 (=1·1). Therefore, even if a less relevant resource has many semantic path instances, it obtains a small relevance score by using Eq. (6). Rocha et al. (2004) and Stojanovic et al. (2003) also reflect the specificity of a given path to determine the relevance of the path, but the measures are different from ours. Rocha et al. (2004)'s specificity is inversely proportional to the in-degree of the object for a given property while ours is inversely proportional to the out-degree of the subject. Stojanovic et al. (2003)'s one is inversely proportional to the multiplication of the out-degree of the subject and the in-degree of the object for a given property.

Basically, our relevance model is based on the weight of the semantic path determined by the information theoretic framework, and in order to solve the occasional side effect caused by the presence of many meaningless semantic path instances, the heuristic using the specificity of semantic path instances is applied.

5.2.2. The coverage of keywords

Generally, a user prefers results covering all query keywords. To reflect this preference, we design a method to measure the coverage of query keywords which is derived from the extended boolean model (Salton et al., 1983).

For a resource a in the set of results A and the set of query keywords K, we map a to a point in a $|K|$-dimensional space $[0,1]^{|K|}$. Each coordinate of a represents the relevance of a to the corresponding keyword. The relevance for each keyword is measured by Eq. (6). The relevance of a for K is in inverse proportion to the distance from the ideal position $[1,\ldots,1]$ to the point of a. The relevance considering the keyword coverage $Cov(a,K)$ is computed by

$$Cov(a,K) = 1 - \left[\frac{\sum_{1\leqslant i\leqslant|K|}(1-NR(a,k_i))^p}{|K|}\right]^{\frac{1}{p}} \tag{7}$$

$$\textit{where } NR(a,k_i) = \frac{R(a,k_i)}{\max_{a_m\in A}R(a_m,k_i)} \quad \text{for} \quad k_i \in K$$

We use the L_p distance and normalize the value into $[0,1]$. $NR(a,k_i)$ represents the normalized relevance of a about k_i. p ($\geqslant 1$) is a tunable parameter which controls the strength of AND-semantics. As p increases, Eq. (7) enforces the AND-semantics among keywords. On the other hand, when $p = 1$, Eq. (7) represents OR-semantics (Salton et al., 1983).

In the IR literature, the cosine similarity is a popular method to measure the similarity between two vectors of n dimensions. In our problem, the query vector consists of the weights of query keywords, and an element of the resource vector is the relevance of the resource for the corresponding query keyword. The overall relevance of the resource for all query keywords can be computed by the cosine similarity between the two vectors. The cosine similarity is determined by the cosine of the angle between the two vectors. Therefore, the sizes of the vectors are not reflected in the similarity. However, in our problem, the difference of the vector magnitudes should be considered. Thus, the cosine similarity is not a suitable alternative for combining multi-keyword relevances.

5.2.3. The discriminating power of keyword

A resource having semantic paths to discriminating keywords is more relevant than a resource having semantic paths to undiscriminating keywords.

The discriminating power of a keyword is measured by the keyword's inverse resource frequency (*irf*) which is similar to *idf* (Salton and McGil, 1986). The inverse resource frequency of a keyword k_i is computed as below:

$$irf(k_i) = \log\frac{|DV|}{|DV_{k_i}|}$$

where $|DV|$ denotes the total number of data values and $|DV_{k_i}|$ is the number of data values containing keyword k_i.

We define the discriminating power of a keyword k_i, $D(k_i)$, as the normalized value of $irf(k_i)$.

$$D(k_i) = \frac{irf(k_i)}{\max_{k_m\in K}irf(k_m)} \tag{8}$$

The discriminating power of a keyword in Eq. (8) represents the weight of a keyword reflecting the importance of the keyword to determine the relevance. Thus, to make the final rank formula, we combine $D(k_i)$ in Eq. (8) to $Cov(r,K)$ in Eq. (7) as follows:

$$Rank(a,K) = 1 - \left[\frac{\sum_{1\leqslant i\leqslant|K|}(D(k_i)\cdot(1-NR(a,k_i)))^p}{\sum_{1\leqslant i\leqslant|K|}D(k_i)^p}\right]^{\frac{1}{p}} \tag{9}$$

Recall that the number of meaningful semantic path instances is reflected to Eq. (7). Consequently, the three factors mentioned in Section 3.2 are fully reflected in Eq. (9).

5.2.4. An example of ranking query results

The semantic path instances shown in Fig. 2(b) are summarized in Table 1. We assume that the properties in Table 1 have the weights as presented in Table 2. The weights of the semantic paths in Table 3 are obtained based on the weights of the properties in Table 2.

Table 1
Semantic path instances from $prof_1$ to data values.

ID	Semantic path instances
ip_1	*interestedIn* ($prof_1$, $topic_1$) *hasName* ($topic_1$, 'Web Search')
ip_2	*writtenBy*$^{-1}$($prof_1$, pub_1) *hasTitle* (pub_1, 'top-k... Web')
ip_3	*writtenBy*$^{-1}$($prof_1$, pub_1) *cite* (pub_1, pub_3)
ip_4	*writtenBy*$^{-1}$($prof_1$, pub_2) *hasTitle* (pub_2, '... Semantic Web')
ip_5	*hasAdvisor*$^{-1}$($prof_1$, st_1) *writtenBy*$^{-1}$(st_1, pub_2)*hasTitle* (pub_2, '... Semantic Web')

Table 2
The weights of properties.

Property	Weight
interestedIn (Person, Topic)	0.5
writtenBy^{-1}(Author, Publication)	0.7
hasName (Topic, String)	0.9
hasTitle (Publication, String)	1
cite (Publication, Publication)	0.4
hasAdvisor (Professor, Student)	0.5

Table 3
The weights of semantic paths ($\delta = 0.6$).

ID	Semantic path	Weight
sp_1	*interestedIn (Person,Topic) hasName (Topic, String)*	0.27
sp_2	*writtenBy^{-1} (Author, Publication) hasTitle (Publication, String)*	0.42
sp_3	*writtenBy^{-1} (Author, Publication) cite (Publication, Publication) hasTitle (Publication, String)*	0.1008
sp_4	*hasAdvisor (Professor, Student) writtenBy^{-1}(Author, Publication) hasTitle (Publication, String)*	0.126

Let's obtain the relevance score of $prof_1$ for semantic query $\langle Professor, \{\text{'top-k'}, \text{'Web'}\}\rangle$. $prof_1$ has five semantic path instances from ip_1 to ip_5 as shown in Table 1. The relevance score of $prof_1$ is obtained by the following process:

(1) The relevance score for each keyword is calculated by Eq. (6).

$$R(prof_1, \text{'top-k'}) = W(ip_2) \cdot spec(ip_2) + W(ip_3) \cdot spec(ip_3) = 0.42 \cdot \left(\frac{1}{2} \cdot 1\right) + 0.1008 \cdot \left(\frac{1}{2} \cdot 1 \cdot 1\right) = 0.2604$$

$$\begin{aligned} R(prof_1, \text{'Web'}) &= W(ip_1) \cdot spec(ip_1) + W(ip_2) \cdot spec(ip_2) + W(ip_4) \cdot spec(ip_4) + W(ip_5) \cdot spec(ip_5) \\ &= 0.27 \cdot (1 \cdot 1) + 2 \cdot 0.42 \cdot \left(\frac{1}{2} \cdot 1\right) + 0.126 \cdot (1 \cdot 1 \cdot 1) = 0.816. \end{aligned}$$

Assume that the maximum $R(r_m, \text{'top-k'})$ is 0.6 and the maximum $R(r_m, \text{'Web'})$ is 0.9. Then, $NR(prof_1, \text{'top-k'}) \approx 0.43 (= \frac{0.2604}{0.6})$ and $NR(prof_1, \text{'Web'}) \approx 0.9 (= \frac{0.816}{0.9})$.

(2) When the inverse resource frequency of each keyword is $irf(\text{'top-k'}) = 11.5$ and $irf(\text{'Web'}) = 5.7$, the discriminating power of each keyword is $D(\text{'top-k'}) = 1$ and $D(\text{'Web'}) \approx 0.5$.

(3) Based on the relevance for each keyword and the discriminating power of the keywords, the final relevance score of $prof_1$ is calculated by Eq. (9) with $p = 3$:

$$Rank(prof_1, K) = 1 - \left[\frac{(D(\text{'top-k'}) \cdot (1 - NR(prof_1, \text{'top-k'})))^3 + (D(\text{'Web'}) \cdot (1 - NR(prof_1, \text{'Web'})))^3}{D(\text{'top-k'})^3 + D(\text{'Web}\prime)^3}\right]^{\frac{1}{3}} \approx 0.45.$$

6. Enhancement in efficiency of semantic search

In this section, we explain how to improve the efficiency in the semantic relationship extraction and the *top-k* result retrieval.

6.1. Semantic relationship extraction

As mentioned in Section 4, the semantic relationship extraction module decides the semantic paths to be examined in the search process. At first, we should know the data properties whose values contain some keywords in *K*. To do this, we construct the table *IT* (*term*, *property*, *class*), where *term* is a keyword, *property* is a property whose object contains *term*, and *class* is the class of *propety*'s subject. We can get the semantic paths from *T* to a given keyword by the breadth-first search of the schema staring from *T* to the extracted set of (*class*, *property*) pairs for the keyword. Table 4 shows the traversal of the schema in Fig. 2(a) for the query $\langle Professor, \{\text{'Web'}\}\rangle$. In addition, since a class inherits the properties of its super classes, the super classes of *T* (i.e., *Person*) are considered in the schema traversal. If an instance is defined as multiple classes, the instance can have all properties of those classes. Thus, if a class contains common instances with *T*, the semantic paths from the class should be considered. We assume that some instances are defined as both *Author* and *Professor*. Finally, the extracted semantic paths are $sp_{2.1}$, $sp_{2.2}$, $sp_{3.1}$, and $sp_{3.2}$.

A lot of semantic paths can be extracted in this step. Besides, if there is a semantic path in which the source class is the same as the destination class (i.e., a cycle), the extraction is not finished. Therefore, some constraints are required.

Length Threshold: As mentioned earlier, long semantic paths are relatively less important to determine the relevance of a result than short paths. Hence, we restrict the length of semantic paths by using a threshold TH_l. The value of the threshold TH_l is dependent on the domain.

Table 4
An example of the semantic path enumeration.

Iter	ID	Semantic path	From
1st	$sp_{1.1}$	*interestedIn (Person, Topic)*	
	$sp_{1.2}$	*writtenBy*$^{-1}$*(Author, Publication)*	
	$sp_{1.3}$	*hasAdvisor*$^{-1}$*(Professor, Student)*	
2nd	$sp_{2.1}$	*interestedIn (Person, Topic) hasName (Topic, String)*	$sp_{1.1}$
	$sp_{2.2}$	*writtenBy*$^{-1}$*(Author, Publication) hasTitle (Publication, String)*	$sp_{1.2}$
	$sp_{2.3}$	*writtenBy*$^{-1}$*(Author, Publication) cite (Publication, Publication)*	$sp_{1.2}$
	$sp_{2.4}$	*hasAdvisor*$^{-1}$*(Professor, Student) writtenBy*$^{-1}$*(Author,Publication)*	$sp_{1.3}$
3rd	$sp_{3.1}$	*writtenBy*$^{-1}$*(Author, Publication) cite (Publication, Publication)hasTitle (Publication, String)*	$sp_{2.3}$
	$sp_{3.2}$	*hasAdvisor*$^{-1}$*(Professor, Student) writtenBy*$^{-1}$*(Author, Publication)hasTitle (Publication, String)*	$sp_{2.4}$

Weight Threshold: Even after pruning with the length of the semantic path, many meaningless semantic paths may remain. The weight of a semantic path presents the contribution of the semantic path to determine the relevance. Therefore, we can prune the semantic paths with a weight smaller than a threshold TH_w.

The weight threshold should be determined according to the data set. Thus, we reflect the weights of properties in each semantic path to TH_w. Since the weight of a semantic path is computed by Eq. (4), we make the weight threshold TH_w based on Eq. (4).

TH_w is computed by aggregating the weights of properties in each position of semantic paths. Many semantic paths tend to have a common suffix since the types of properties having a particular keyword are limited. Therefore, to compute TH_w, we align the properties in semantic paths in inverse order.

For a semantic path sp_j with $length(sp_j) \leqslant \mathrm{TH}_l$, let $\bar{p}_{ji}$ denote the $(length(sp_j) - i + 1)$th property in sp_j. For example, for a semantic path sp_4 in Table 3, $\bar{p}_{41}$ is *hasTitle*, $\bar{p}_{42}$ is *writtenBy*$^{-1}$, and $\bar{p}_{43}$ is *hasAdvisor*$^{-1}$.

To obtain TH_w, the representative weight tw_i for the $(length(sp_j) - i + 1)$th properties in all semantic paths sp_j such that $length(sp_j) \leqslant TH_l$ is required. The maximum, average, minimum functions can be used to compute tw_i. In this work, we use the maximum function which shows a better performance than others. Thus, tw_i is computed as follows:

$$tw_i = max_{sp_j \in SP} w(\bar{p}_{ji})$$

where $\bar{p}_{ji}$ is the $(\mathrm{length}(sp_j) - i + 1)$th property in sp_j.

Consequently, the weight threshold TH_w is computed as follows:

$$TH_w = \left(\prod_{1 \leqslant i \leqslant TH_l} tw_i \right) \cdot \delta^{TH_l - 1} \tag{10}$$

In case that $max(length(sp_j)) \langle TH_l, tw_i = 1$ for $i \rangle$ max $(length(sp_j))$.

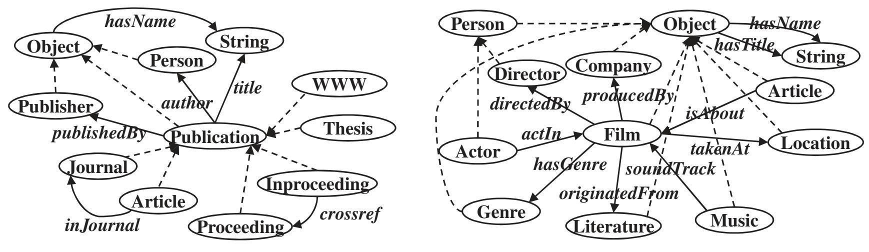
We do not apply the weight threshold TH_w blindly since a unique semantic path to a particular resource type directly related to keywords can be pruned. The removal of this kind of semantic paths can lead to the missing of the relevant result. For example, consider the following three semantic paths:

sp_1 :*writtenBy*$^{-1}$ *(Author, Publication) hasTitle(Publication, String)*
sp_2 :*writtenBy*$^{-1}$*(Author, Publication) cite*$^{-1}$*(Publication, Publication) hasTitle(Publication, String)*
sp_3 :*memberOf (Person, ResearchGroup) performedBy*$^{-1}$*(ResearchGroup, Project) hasName(Project, String)*

The semantic path sp_1 and sp_2 represent the relationships between *Author* and *Publication ('s title)* and sp_3 represents that between *Person* and *Project ('s name)*. If $W(sp_1) > TH_w > W(sp_3) > W(sp_2)$, sp_2 and sp_3 are pruned. In order to find students relevant to a keyword, most publications which should be examined are covered by sp_1. Therefore, the publications reached by sp_2 is ignorable. However, if projects related to the keyword can be covered by only sp_3, many relevant students can be missed by pruning of sp_3. Thus, in order to preserve the important semantic path like sp_3, our approach prunes the semantic paths according the following pruning rule.

Pruning Rule: A semantic path $sp_i(=p_{i1}(d_{i1}, r_{i1}) \cdots p_{in}(d_{in}, r_{in}))$ with a weight less than TH_w is pruned, if there is another semantic path $sp_j(=p_{j1}(d_{j1}, r_{j1}) \cdots p_{jm}(d_{jm}, r_{jm}))$ which satisfies all of the following conditions: (1) $W(sp_i) < W(sp_j)$, (2) $p_{in} = p_{jm}$, and (3)r_{jm-1} is equal to r_{in-1} or a super class of r_{in-1}.

Some resources related to the query can be excluded from the answer through the pruning. However, if a resource is fairly relevant to the query, it is likely to be related to the query keywords through a more important semantic path than the pruned paths. Furthermore, since the weights of the pruned semantic paths are small, the influence of the removal of them is ignorable. Therefore, in general, the accuracy of search is not degraded by the pruning of sematic paths. Moreover, as mentioned in Section 5.1, a semantic path with a small weight tends to contain relatively much more instances than others. Thus, the pruning of such semantic paths effectively reduce the search space.



(a) DBLP (b) IMDB

Fig. 5. Schemas of data sets.

6.2. *top-k* Result retrieval

Although our pruning technique effectively reduces the search space, it is impractical to traverse a huge instance graph to find *top-k* results for every query. In order to efficiently access the *top-k* results, we construct and materialize ‘keyword index’, which is an extended inverted index, through an off-line preprocessing. The keyword index is composed of a set of index lists for keywords, and we can efficiently access the list of resources related to a keyword through the keyword index. An entry of the index list for a keyword is a triple consisting of the type, identifier, and relevance score of a resource related to the keyword through semantic paths. Each index list is sorted in descending order of the relevance score. For the construction of the index list for a keyword, we find the semantic paths from each class to the keyword in the schema of the ontology. Then, we prune meaningless semantic paths among them by using the pruning technique mentioned in the previous section. The construction of the keyword index is a time-consuming task. However, once the keyword index has been constructed, we can directly access the resources related to a particular keyword through the index without the expensive traversal of the instance graph for every query request.

To answer a query $\langle T, \{k_1, \ldots, k_m\}\rangle$, we first access the index list for every keyword from the keyword index. Then, we find the *top-k* results having the highest overall relevance scores. The overall score is computed by Eq. (9). The naive algorithm performs full scans of m index lists and computes the overall scores of all accessed resources. Then, it extracts k resources with the highest overall scores. It is inefficient for large sized index lists because the cost is proportional to the size of the index lists. Therefore, we adapt Threshold Algorithm (TA) (Fagin et al., 2003) which is a representative algorithm to efficiently retrieve *top-k* results from multiple sorted index lists. Basically, this algorithm sequentially accesses resources and their relevance scores for each keyword in the index lists. In other words, the resources are accessed in descending order of the relevance score, thereby the resources with sufficient possibility to have a higher overall score are retrieved earlier than others on the assumption that the overall scoring function is monotonic. At the same time, in order to immediately compute the overall relevance score for each accessed resource, it can perform random accesses to obtain unseen local scores for the resource. The process is terminated when no resource having a higher score than the minimum score of the *top-k* resources retrieved so far can be accessed by further scan. In addition, if T is not the root class of the class hierarchy, resources which do not belong to T are filtered out. Thus, scanning the entire index lists can be avoided.

7. Experiments

In the experiments, we evaluate the accuracy of our search method in comparison with existing methods by using real data sets. In addition, we observe the sensitivity of our ranking method for tunable parameters, and validate the effectiveness of our pruning method. Finally, we evaluate the efficiency of the *top-k* retrieval using the keyword index.

7.1. Data sets and queries

We construct two kinds of ontologies from two real datasets: DBLP data[3] and IMDB data.[4] Fig. 5 shows the schemas of the DBLP and IMDB ontoloiges. The DBLP ontology includes 798,468 class instances and 3,141,309 property instances, and the IMDB ontology contains 2,617,977 class instances and 12,222,558 property instances.

For an effective evaluation of our ranking method, we need ontologies having various classes and properties as well as many instances since some of our measures are based on self-information and mutual information which are defined as functions of probability distribution. In addition, the content of the ontologies should be so familiar with users as to easily judge the relevance of search results. The DBLP and IMDB ontologies satisfy these requirements. Moreover, they have differences in the following aspects which determine the performance of the semantic search: First, IMDB has more various properties than DBLP. Second, the terms in IMDB are more general and implicit than those in DBLP. Due to these differences, the

[3] http://dblp.uni-trier.de/xml/. The DBLP data used in our experiments contains the information of publications published before 2005.

[4] http://www.imdb.com/interface.

size of the semantic search scope and the accuracy of the semantic search for those ontologies are quite different. Therefore, we use the DBLP and IMDB ontologies as data sets in our experiments.

We select 20 queries for DBLP and 15 queries for IMDB with diverse characteristics related to search scope and ranking in order to evaluate our ranking method and show the difference between our approach and other methods. The criteria for selecting queries are the desired resource type, the number of relevant results, the importance of keywords, and the number of keywords ranging from 2 to 5.

Tables 5 and 6 show the queries and the size of their relevant answer (i.e., #RA). It is impossible to scan the entire database to identify all relevant results. Thus, we use the depth-*k*-pooling method in order to determine the set of relevant results, where *k* is 20 in our experimental study. For each query, we first consolidate all *top*-20 results of every search/ranking method. Then, the relevant results (i.e., *RA*) are manually labeled by examining the consolidated results based on the facts and the discussion among researchers of our database group. The basis for judgment to decide the relevance of search results is the user intention for each query, which is indicated by ‘U’, presented in Tables 5 and 6. In the query ID, the prefix ‘D’ denotes queries for DBLP and ‘I’ for IMDB. ‘U’ indicates the user intention about the query, and it is used to judge the relevance of search results.

7.2. Ranking methods

We compare our ranking method with other existing methods which also consider an ontology in a search. One is proposed by Stojanovic et al. (2003), which is indicated by *RQR* in this paper. The other is proposed by Rocha et al., 2004, which is indicated by *HAS*. Additionally, we compare with the basic keyword-based search engines provided in the DBLP and IMDB sites in order to show the effectiveness of our semantic search using an ontology. For a fair comparison, we narrow the search scope in the DBLP

Table 5
DBLP query sets.

ID	DBLP query	$\|RA\|$
DQ1	⟨*article, {‘software’,‘testing’}*⟩ U: Article about software testing	25
DQ2	⟨*person, {‘multimedia’,‘index’,‘database’}*⟩ U: Author who writes publications about indexing on multimedia database	31
DQ3	⟨*inproceeding, {‘xml’,‘index’,‘database’}*⟩ U: Inproceeding about indexing on xml database	30
DQ4	⟨*inproceeding, {‘distributed’,‘query’,‘cost’,‘model’}*⟩ U: Inproceeding about cost model on distributed query processing	22
DQ5	⟨*publication, {‘top-k’,‘relational’,‘database’,‘2002’,‘2003’}*⟩ U: Publication which is about top-k answering on relational database and was published in 2002 or 2003	6
DQ6	⟨*article, {‘semantic’,‘web’,‘service’}*⟩ U: Article about semantic web service	20
DQ7	⟨*inproceeding, {‘stream’,‘data’}*⟩ U: Inproceeding about stream data processing	77
DQ8	⟨*article, {‘spatial’,‘temporal’,‘database’}*⟩ U: Article about spatial–temporal database	17
DQ9	⟨*inproceeding, {‘face’,‘recognition’,‘artificial’,‘intelligence’}*⟩ U: Inproceeding about face recognition in artificial intelligence field	12
DQ10	⟨*inproceedings, {‘content’,‘based’,‘information’,‘retrieval’,‘multimedia’}*⟩ U: Inproceeding about CBIR in multimedia database	35
DQ11	⟨*inproceedings, {‘keyword’,‘search’}*⟩ U: Inproceeding about keyword search	23
DQ12	⟨*article, {‘workflow’,‘management’}*⟩ U: Article about workflow management	34
DQ13	⟨*person, {‘data’,‘mining’,‘outlier’,‘detection’}*⟩ U: Author who writes publications about outlier detection in data mining	15
DQ14	⟨*inproceeding, {‘web’,‘service’}*⟩ U: Inproceeding about web service	48
DQ15	⟨*inproceedings, {‘data’,‘integration’}*⟩ U: Inproceeding about data integration	50
DQ16	⟨*inproceedings, {‘transaction’,‘recovery’,‘database’}*⟩ U: Inproceeding about data transaction recovery on database	44
DQ17	⟨*inproceedings, {‘workflow’,‘management’,‘web’}*⟩ U: Inproceeding about workflow management in web	19
DQ18	⟨*inproceedings, {‘relevance’,‘feedback’,‘multimedia’,‘database’}*⟩ U: Inproceeding about relevance feedback in multimedia database field	23
DQ19	⟨*inproceedings, {‘personalized’,‘web’,‘search’}*⟩ U: Inproceeding about personalized web search	13
DQ20	⟨*inproceedings, {‘xml’,‘update’}*⟩ U: Inproceeding about update in xml database	24

Table 6
IMDB query sets.

ID	IMDB query	$\|RA\|$
IQ1	⟨*director*, {*'shakespeare','hamlet'*}⟩ U: Director who made films about shakespeare's hamlet	34
IQ2	⟨*company*, {*'japan','animation'*}⟩ U: Japanese company which produces animations	29
IQ3	⟨*actor*, {*'romance','comedy'*}⟩ U: Actor who acts in romance-comedy films	44
IQ4	⟨*film*, {*'steven','spielberg','adventure'*}⟩ U: Adventure film directed by steven spielberg	14
IQ5	⟨*film*, {*'jane','austen','pride','prejudice'*}⟩ U: Film originated from 'pride prejudice' of jane austen	19
IQ6	⟨*film*, {*'romeo','juliet'*}⟩ U: Film about romeo and juliet	47
IQ7	⟨*actor*, {*'action','comedy'*}⟩ U: Actor who acts in action-comedy films	19
IQ8	⟨*actor*, {*'usa','musical'*}⟩ U: Actor who acts in musical films made in USA	37
IQ9	⟨*article*, {*'artificial','intelligence'*}⟩ U: Article about Artificial Intelligence (AI)	11
IQ10	⟨*article*, {*'star','wars','attack','clone'*}⟩ U: Article about Star Wars: Attack of the clone	14
IQ11	⟨*film*, {*'lucas','star','wars'*}⟩ U: Star Wars directed by Lucas	4
IQ12	⟨*music*, {*'jesus','christ','superstar','1973'*}⟩ U: Soundtrack of 'jesus christ superstar' produced in 1973	27
IQ13	⟨*film*, {*'tim','burton','thriller'*}⟩ U: Thriller directed by Tim Burton	5
IQ14	⟨*film*, {*'tim','burton','animation'*}⟩ U: Animation directed by Tim Burton	8
IQ15	⟨*film*, {*'johnny','depp','fantasy'*}⟩ U: Fantasy film in which Johnny Depp acts	4

and IMDB search whenever possible in place of the type constraint of a query in our semantic search. We use the 'refined by type option in the DBLP search engine, and we use a list box to select the search scope in the IMDB search engine.

The DBLP data used in our experiments is not up-to-date. Therefore, for the queries on the DBLP data, in order to conduct a fair comparison, we consider publications published before 2005 among the results returned from the DBLP search engine.

HAS has a weight factor which reflects the semantics of a relationship, and the weights of all properties are determined by domain experts. Since we have no basis for judgment to decide the weights, we determine the semantic weights of properties in *HAS* by using our weighting measure for properties (i.e., Eq. (3)).

Our ranking method is represented by *SSR*. In addition, we evaluate the effectiveness of the pruning by the weight threshold (TH_w) according to the pruning rule in Section 6. *SSR-T* denotes *SSR* applying the pruning.

We perform the experiments on AND-semantics. Other experimental environments such as the storage of the ontology and the semantic search process are common. In order to restrict the search space, we use $TH_l = 3$.

7.3. Metrics

In order to measure the accuracy of ranking methods, we observe the *top*-10 and *top*-20 results for each query. We use the following well-known metrics in the information retrieval field (Manning et al., 2008):

- Precision: $P = \frac{|A \cap RA|}{|A|}$, where A is the set of retrieved results and RA is the set of relevant answers.
- Recall: For *top-k* answering, the size of the entire set of relevant results is not important when it contains more than k results. Also, it is difficult to find out the total size of the answer set for a query in a large database. Thus, when the size of the relevant results discovered in our experiments is larger than k, we regard k as the size of the relevant results and compute recall as follows: If —RA— < k, the recall $R = \frac{|A \cap RA|}{|RA|}$. Otherwise, $R = \frac{|A \cap RA|}{k}$.
- F-measure: We use the harmonic mean of precision and recall, $F\text{-}Score = \frac{2PR}{P+R}$.
- Mean Average Precision: $MAP = \frac{\sum_{q=1}^{|Q|} AveP(q)}{|Q|}$, where $AveP = \frac{\sum_{k=1}^{20} (P(k) \times rel(k))}{|RA|}$, $|Q|$ is the number of queries, $P(k)$ is the precision at cut-off k, and $rel(k)$ is 1 if the kth result is a relevant one, zero otherwise.

Additionally, in order to measure the effectiveness of pruning the search space, we count the number of semantic paths and the number of resources traversed by *SSR* and those by *SSR-T*.

Finally, in order to measure the efficiency of the *top-k* answering using the keyword index, we observe the query response time for the *top-k* answering for each query.

7.4. Experiment results

7.4.1. Accuracy of ranking method

Tables 7–10 present the accuracy (i.e., Precision, Recall, F-Score, and MAP) of four ranking methods and the baseline search engines. The tables include the five representative queries and the full results are shown in Appendix A. We also include the average of the accuracy for 20 queries over DBLP (i.e., AVG20) and 15 queries over IMDB (i.e., AVG15) in the tables. As the values of the tunable parameters in *SSR (-T)*, we use 0.2 for α, 0.8 for β in Eq. (3), 0.6 for δ in Eq. (4), and 3 for p in Eq. (9).

We can see the effectiveness of the semantic search using an ontology by comparing the results with those of the DBLP and IMDB search engines. The improvement of the accuracy is mainly due to the consideration of the semantic relationships between the target resources and the query keywords in searching and ranking. The semantic search method can retrieve resources that contain the query keywords in their textual descriptions as well as resources that are indirectly associated with the query keywords through semantic paths. In contrast, the baseline search engines do not consider the indirect associations between the query keywords and the resources. In case of the DBLP database, most of the key information of a publication (i.e., the title of the publication, the authors, the name of the conference/journal, and the publication date) is represented in a single text of the publication. Thus, the basic keyword-based search is sufficient to retrieve the relevant results for many queries, thereby the performance of the DBLP search engine is not worse compared to that of semantic search methods. On the other hand, in the IMDB database, the indirect semantic relationships among resources play an important role in finding relevant results. As a result, the performance of the IMDB search engine is much worse than that of the semantic search methods, except queries finding films using the keywords in their titles (i.e., IQ5 and IQ6). For example, for IQ3, the semantic search methods access the answer (i.e., an actor who takes part in romantic-comedy films) through the semantic path '*actIn (Actor, Film) hasGenre (Film, Genre)*' while the IMDB search engine misses many relevant actors whose biographies do not contain the query keywords in the IMDB database. In our experiments, SSR (-T) achieves 6.9–16.7% improvement in the accuracy (i.e., F-Score) for DBLP and 49.5–58.2% improvement for IMDB.

The proposed weighting measure for semantic paths is designed to reflect the different importance of the paths to discriminate the relevant resources from others. *SSR (-T)* and *HAS* using our weighting measure take the importance of each semantic path into account in ranking resources. Consequently, *SSR (-T)* and *HAS* outperform *RQR* in general. In addition, since *SSR (-T)* considers more relevance criteria and combines them effectively than *RQR* and *HAS*, *SSR (-T)* usually outperforms them. In summary, *SSR (-T)* provides 11.6–22.1% improvement for DBLP and 14.6–24.5% for IMDB compared with *RQR* and 8.5–21.8% improvement for DBLP and 4.8–15.8% for IMDB compared with *HAS*. Furthermore, we consider the order in which the returned results are presented. Table 11 shows the mean of the average precision scores for each query (MAP). We can see that SSR (-T) tends to present the relevant results in higher positions than other semantic search methods.

In the aspects of the keyword coverage and the discriminating power of keywords, *HAS* complies with the strict AND-semantics so that it returns only results covering all query keywords. *RQR* simply multiplies the relevance score of a resource by the number of keywords related to the resource. Also, they do not reflect the importance of each keyword. In contrast, *SSR (-T)* uses an adapted extended boolean model weighted by the discriminating power of keywords. Thus, *SSR (-T)* is more effective compared with other ranking methods for queries, where the number of keywords is large and the keywords with high discriminating power play an important role to determine the relevance. For example, in case of *DQ*5 including an OR-semantics (i.e., 2002 or 2003), *HAS* cannot return any results. In addition, the *irf*s of '*top-k*' and '*relational*' are 11.6 and 6.1 while the *irf*s of other keywords are around 3. In fact, '*top-k*' and '*relational*' are more important than other keywords to

Table 7
The accuracy of *top*-10 DBLP query results.

Metric	Query	RQR	HAS	SSR	SSR-T	DBLP-Engine
Precision	DQ1	1	1	1	1	1
	DQ2	0.1	0.2	0.5	1	1
	DQ5	0.3	0	0.6	0.6	0.2
	DQ9	0.5	0.5	0.5	0.8	0.2
	DQ18	0.7	0.8	0.9	0.9	0.2
	AVG20	0.675	0.68	0.805	0.895	0.73
Recall	DQ1	1	1	1	1	1
	DQ2	0.1	0.2	0.5	1	1
	DQ5	0.375	0	0.75	0.75	0.25
	DQ9	0.5	0.5	0.5	0.8	0.2
	DQ18	0.7	0.8	0.9	0.9	0.2
	AVG20	0.678	0.68	0.812	0.902	0.732
F-Score	DQ1	1	1	1	1	1
	DQ2	0.1	0.2	0.5	1	1
	DQ5	0.33	0	0.66	0.66	0.22
	DQ9	0.5	0.5	0.5	0.8	0.2
	DQ18	0.7	0.8	0.9	0.9	0.2
	AVG20	0.676	0.68	0.808	0.898	0.731

Table 8
The accuracy of *top*-20 DBLP query results.

Metric	Query	RQR	HAS	SSR	SSR-T	DBLP-Engine
Precision	DQ1	1	0.95	1	1	1
	DQ2	0.05	0.35	0.55	1	1
	DQ5	0.15	0	0.3	0.3	0.1
	DQ9	0.55	0.5	0.5	0.55	0.1
	DQ18	0.7	0.6	0.75	0.75	0.1
	AVG20	0.59	0.625	0.702	0.745	0.645
Recall	DQ1	1	0.95	1	1	1
	DQ2	0.05	0.35	0.55	1	1
	DQ5	0.375	0	0.75	0.75	0.25
	DQ9	0.916	0.833	0.833	0.916	0.166
	DQ18	0.7	0.6	0.75	0.75	0.1
	AVG20	0.633	0.654	0.758	0.803	0.667
F-Score	DQ1	1	0.95	1	1	1
	DQ2	0.05	0.35	0.55	1	1
	DQ5	0.214	0	0.428	0.428	0.142
	DQ9	0.687	0.625	0.625	0.687	0.125
	DQ18	0.7	0.6	0.75	0.75	0.1
	AVG20	0.606	0.637	0.722	0.766	0.653

Table 9
The accuracy of *top*-10 IMDB query results.

Metric	Query	RQR	HAS	SSR	SSR-T	IMDB-Engine
Precision	IQ2	0.7	0.9	0.9	0.9	0.6
	IQ3	1	0.8	1	1	0.4
	IQ4	0.1	0.1	0.3	0.9	0
	IQ5	0.8	0.5	0.8	0.9	0.8
	IQ12	0	0.9	0.9	1	0
	AVG15	0.548	0.631	0.696	0.778	0.227
Recall	IQ2	0.7	0.9	0.9	0.9	0.6
	IQ3	1	0.8	1	1	0.4
	IQ4	0.1	0.1	0.3	0.9	0
	IQ5	0.8	0.5	0.8	0.9	0.8
	IQ12	0	0.9	0.9	1	0
	AVG15	0.663	0.741	0.835	0.908	0.228
F-Score	IQ2	0.7	0.9	0.9	0.9	0.6
	IQ3	1	0.8	1	1	0.4
	IQ4	0.1	0.1	0.3	0.9	0
	IQ5	0.8	0.5	0.8	0.9	0.8
	IQ12	0	0.9	0.9	1	0
	AVG15	0.564	0.651	0.723	0.809	0.227

determine the relevance. However, Both *HAS* and *RQR* do not reflect this difference. For such queries, *SSR (-T)* outperforms *RQR* as well as *HAS*.

In addition, after pruning, the accuracy of a semantic search is improved as shown in the results of *SSR* and *SSR-T* in Tables 7–10. Even if the influence of a less important semantic path is attenuated by reflecting its small weight to the relevance score, we should not underestimate the effect of multiple such semantic paths. For instance, for *IQ*3, more relevant films are superseded by irrelevant films. The irrelevant films are indirectly related to many films with titles including the keywords of *IQ*3 through several less meaningful semantic paths such as *'hasGenre (Film, Genre) hasGenre*$^{-1}$*(Genre, Film) hasTitle (Film, String)'*. *SSR-T* effectively prunes such semantic paths and provides the best accuracy among all ranking methods for most of queries.

We perform *t*-test (Hull, 1993) to decide the statistical significance of the improvement by our method *SSR-T* over other semantic search methods *RQR* and *HAS*. If *p*-value is less than 0.05, one can conclude that the improvement is statistically significant. The *t*-test calculates *p*-values based on the accuracy (i.e., F-Score) of *SSR-T* and other methods. For DBLP queries, the *p*-values on the improvement of *SSR-T* over other methods are less than 0.005. For IMDB queries, the *p*-values of *SSR-T* over other methods are less than 0.05. We can conclude that the improvement of the accuracy by *SSR-T* in general is statistically significant over all baseline approaches.

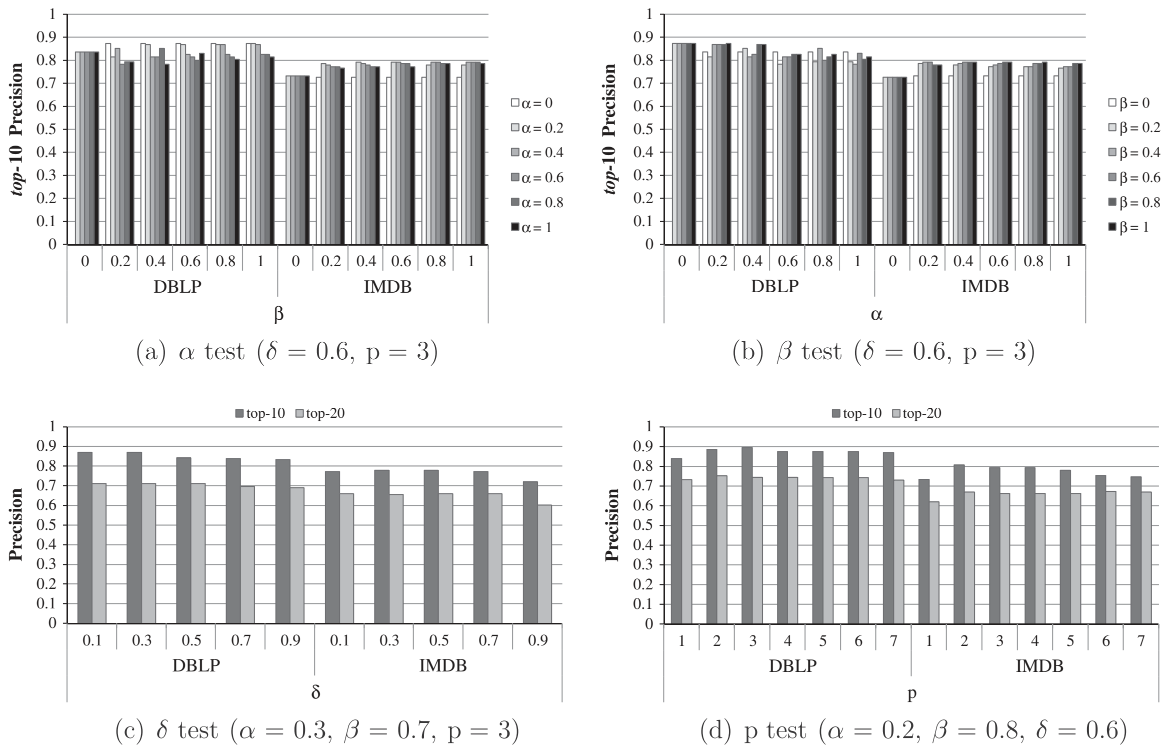
The majority of data values in the DBLP ontology are the titles of publications. In general, a publication title are important terms which can most clearly describe the publication. Thus, we use words which can be found in the titles as query keywords. As a result, in most cases, the search scope (i.e., semantic paths) to find the answer for a query is obvious. On the other hand, the terms in the IMDB ontology are general words, often used in our ordinary life, and even implicit.

Table 10
The accuracy of *top*-20 IMDB query results.

Metric	Query	RQR	HAS	SSR	SSR-T	IMDB-Engine
Precision	IQ2	0.8	0.95	0.95	0.9	0.3
	IQ3	0.75	0.75	1	1	0.2
	IQ4	0.1	0.15	0.15	0.7	0
	IQ5	0.4	0.3	0.4	0.9	0.5
	IQ12	0.15	0.95	0.95	1	0
	AVG15	0.46	0.55	0.59	0.663	0.153
Recall	IQ2	0.8	0.95	0.95	0.9	0.3
	IQ3	0.75	0.75	1	1	0.2
	IQ4	0.142	0.214	0.214	1	0
	IQ5	0.421	0.315	0.421	0.947	0.526
	IQ12	0.15	0.95	0.95	1	0
	AVG15	0.641	0.762	0.842	0.933	0.16
F-Score	IQ2	0.8	0.95	0.95	0.9	0.3
	IQ3	0.75	0.75	1	1	0.2
	IQ4	0.117	0.176	0.176	0.823	0
	IQ5	0.41	0.3	0.41	0.92	0.51
	IQ12	0.15	0.95	0.95	1	0
	AVG15	0.503	0.602	0.65	0.731	0.155

Table 11
MAP for *top-k* query results.

Data set	*top-k*	RQR	HAS	SSR	SSR-T
DBLP	*top*-10	0.25	0.27	0.32	0.38
	top-20	0.39	0.43	0.50	0.59
IMDB	*top*-10	0.31	0.36	0.41	0.45
	top-20	0.40	0.52	0.56	0.65



(a) α test ($\delta = 0.6$, p = 3)

(b) β test ($\delta = 0.6$, p = 3)

(c) δ test ($\alpha = 0.3$, $\beta = 0.7$, p = 3)

(d) p test ($\alpha = 0.2$, $\beta = 0.8$, $\delta = 0.6$)

Fig. 6. Sensitivity test (ranking method: *SSR-T*).

Table 12
The reduction of search space.

Query	# SP	# N	# PSP	# PN	R (%)
DQ1	25	164,958	4	42,693	74.11
DQ2	11	104,970	2	64,390	38.65
DQ5	97	676,549	5	114,422	83.08
DQ9	30	246,320	7	160,788	34.72
DQ18	30	239,614	7	153,490	35.94
AVG20	–	–	–	–	42.80
IQ2	12	38,779	11	37,698	2.78
IQ3	11	422,907	10	422,592	0.07
IQ4	20	663,236	11	61,724	90.69
IQ5	19	583,297	10	12,047	97.934
IQ12	11	160,679	10	147,016	8.50
AVG15	–	–	–	–	44.03

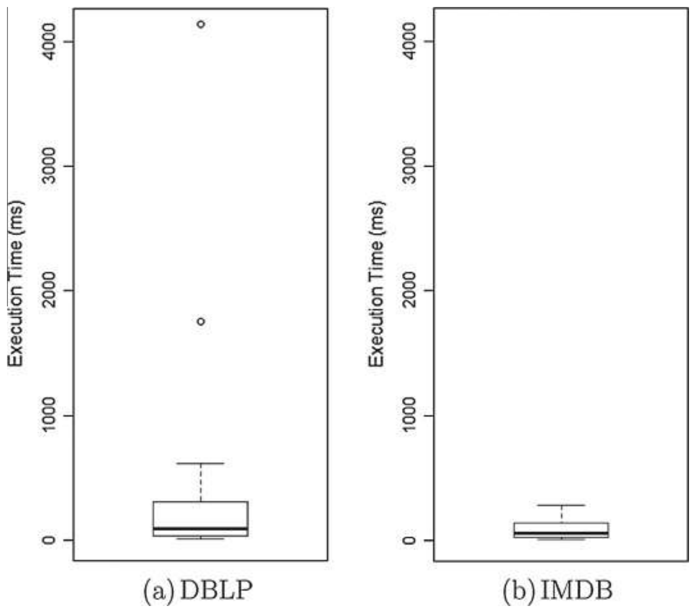


Fig. 7. Execution time for *top*-10 answering.

The keywords in queries for IMDB are found in very diverse locations like movie titles, literature titles, music titles, actor names, genre names, and company names. As a result, it is hard to distinguish the important part which should be searched to find the answer. Consequently, the overall performance for DBLP is better than IMDB.

7.4.2. Sensitivity study

SSR (-T) uses several tunable parameters. Therefore, we observe the accuracy according to the values of the parameters. Figs. 6(a) and (b) shows the accuracy of *SSR-T* for *top*-10 queries with varying α and β in Eq. (3). As the value of α decreases and the value of β increases, the performance tends to be improved. It means that the mutual information factor (MI) is more important to determine the weight of a property than the discriminating power factor (I).

There can be the following side effects for very small or very large δ in Eq. (4). First, if δ is very small, the weight of a long but important semantic path becomes too small to have an effect on the relevance score. Second, if δ is very large, the weight of a long and less important semantic path can get similar or even larger weight than short but important one. Besides, this situation sometimes incurs that the important short path can be pruned by the longer one. These two problems result in the decrease of the accuracy. In case of queries over DBLP, a few short semantic paths are very important to determine the relevance, and most long paths are ignorable. Thus, a very large value of δ has influence on the degradation of the accuracy as in DBLP of Fig. 6(c). In contrast, in case of queries over IMDB, most semantic paths are important to determine the relevance. Thus, the accuracy of some queries decreases due to a small value of δ as in IMDB of Fig. 6(c).

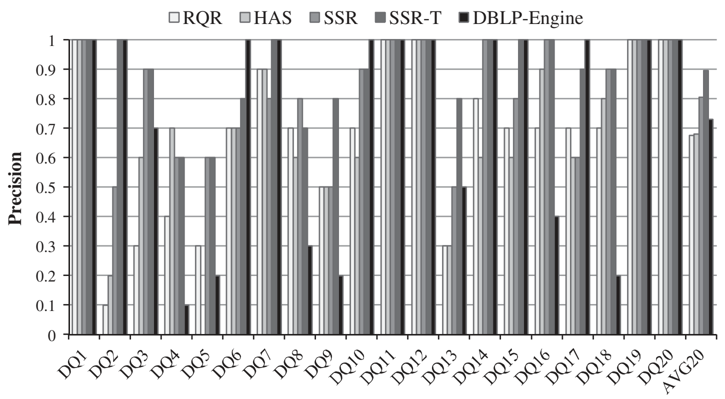
Fig. 6(d) shows the accuracy according to the value of p in Eq. (9). The value of p controls the strength of AND-semantics, and we observe that the p value from 2 to 4 is sufficient to enforce the AND-semantics for most of queries in our experiments.

7.4.3. The reduction of search space

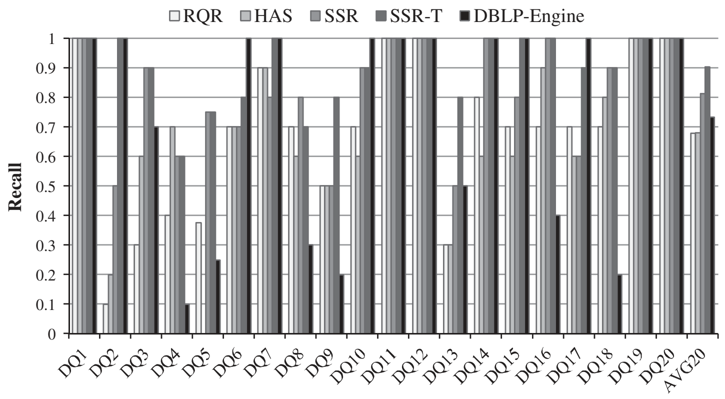
Table 12 shows the effectiveness of the pruning in terms of the reduction of the search space. # SP and # PSP denote the number of semantic paths to be examined by *SSR* and that by *SSR-T*, respectively. # N and # PN indicate the number of resources to be traversed by *SSR* and that by *SSR-T*, respectively. *R* is the reduced portion of the total search space.

In case of the DBLP ontology, the location of keywords are very restricted and almost all properties are relationships between publications or between a publication and an author. This feature makes a large number of semantic paths including a cycle from *Publication* to *Publication* in the schema level, and the common suffixes are replicated in those semantic paths. Therefore, a large part of semantic paths is pruned and it can be avoided to traverse a huge number of semantic path instances.

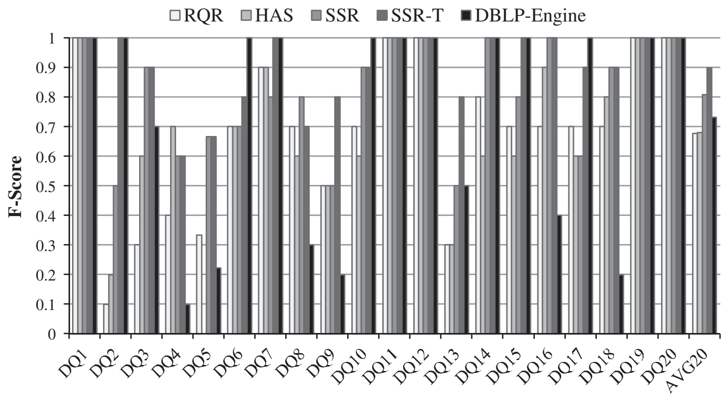
In case of IMDB ontology, most semantic paths are unique paths to a particular resource type directly related to keywords. Thus, the ratios of pruned semantic paths and the resources are smaller than those of DBLP. However, for some queries, the



(a) Top-10 Precision



(b) Top-10 Recall



(c) Top-10 F-Score

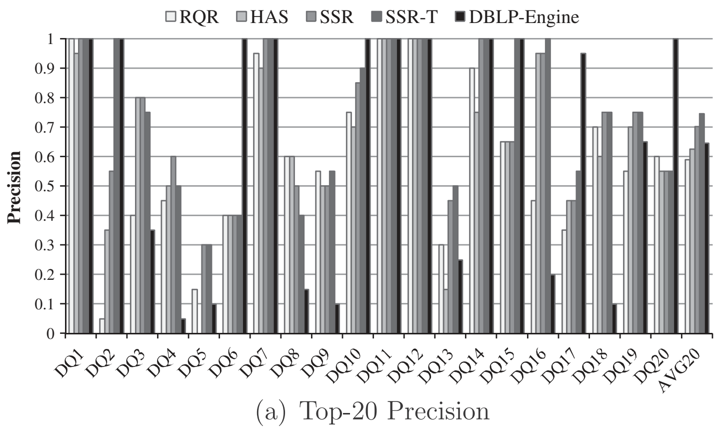
Fig. 8. The accuracy for *top*-10 DBLP query results.

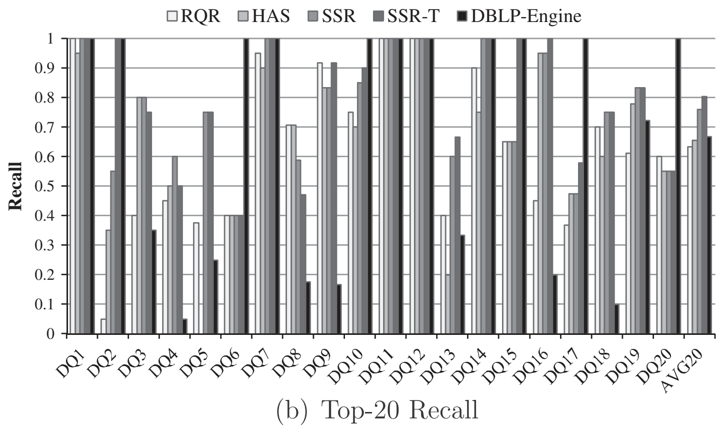
pruning is still effective. For example, in case of *IQ*3 and *IQ*4, most of search space (over 90%) can be removed since the pruned semantic paths such as *'hasGenre (Film, Genre) hasGenre^{-1}(Genre, Film) hasTitle (Film, String)'* include a lot of path instances.

In conclusion, the pruning technique reduces the search space very effectively. However, the pruned semantic paths do not have significant influence on the relevance. Thus, as you can see in Tables 7–10, the accuracy is not degraded but even improved.

7.4.4. *top-k* Retrieval cost

In order to examine the efficiency of the *top-k* retrieval using Threshold Algorithm, we observe the total elapsed time for *top-k* answering for each query. Experiments are conducted on an Intel (R) Pentium (R) 2.60 GHz CPU and 2 GB RAM running





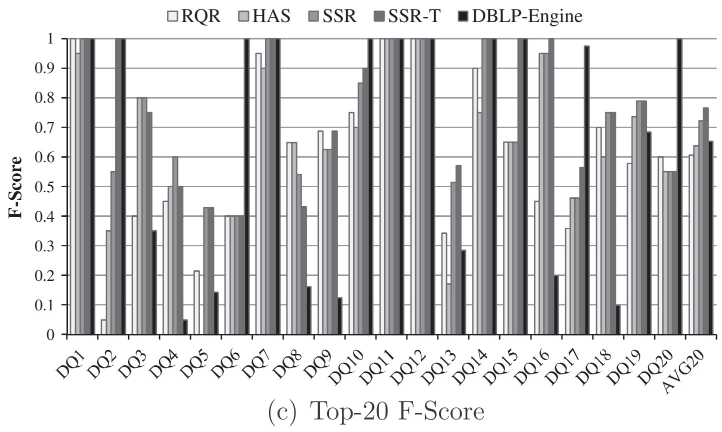
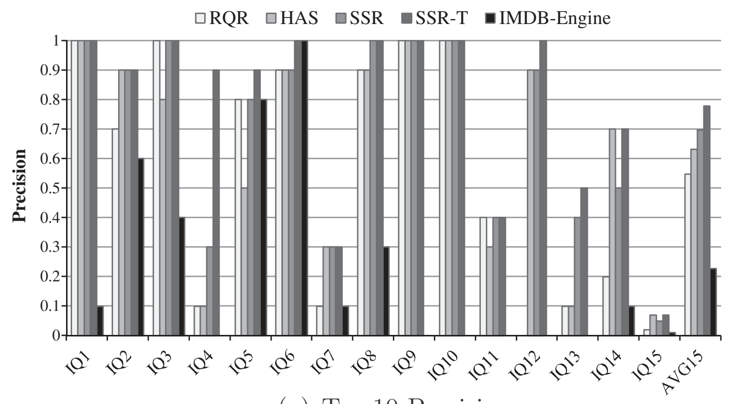
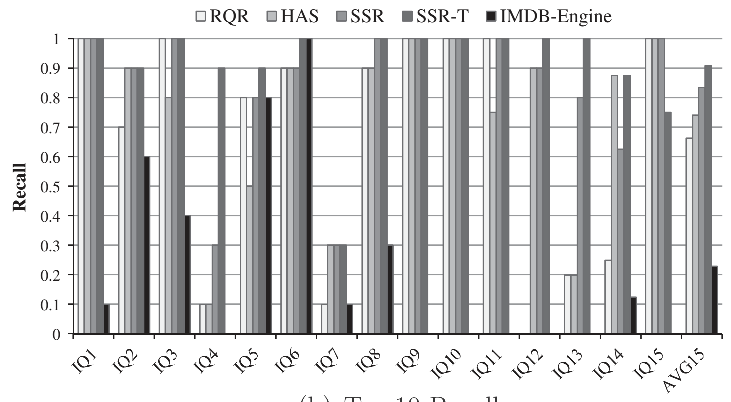


Fig. 9. The accuracy for *top*-20 DBLP query results.

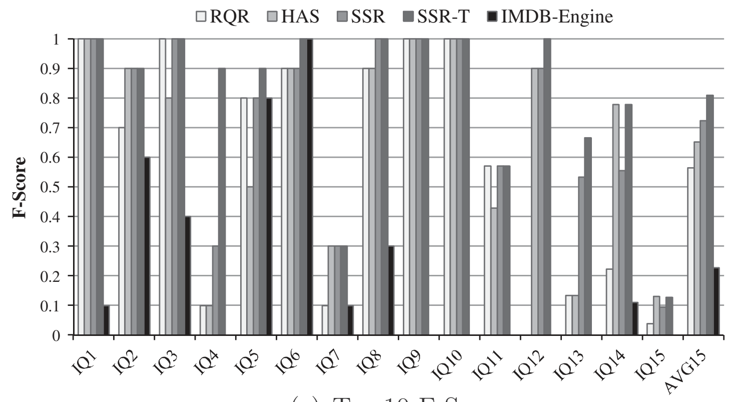
Window XP, and all algorithms are implemented in Java using JBuilder. We set the size of the virtual memory for the execution to 512 MB. We summarize the execution time of *top*-10 answering for all DBLP and IMDB queries as a box plot, which consists of the minimum value, lower quartile, median, upper quartile, and the maximum value, in Fig. 7. For the most of queries, the *top*-10 results are retrieved within a reasonable time less than 0.5 s. Because our *top-k* retrieval using Threshold Algorithm directly accesses the candidate results related to query keywords via the keyword index, the execution time depends on the number of candidate results. Therefore, even though the DBLP dataset is smaller than the IMDB dataset, the execution times for DBLP queries having larger candidate lists (i.e., index list for each keyword) are somewhat greater than those for IMDB queries. In addition, if there are many resources having similar relevance scores, the Threshold Algorithm should examine a lot of resources in the index lists. It is hard to reach the termination condition preserving that no more following resource will have larger relevance score. For example, query DQ2 takes much time (i.e., about 4 s) since it needs to access resources about nine times greater than the average number of accessed resources for all DBLP queries.



(a) Top-10 Precision



(b) Top-10 Recall

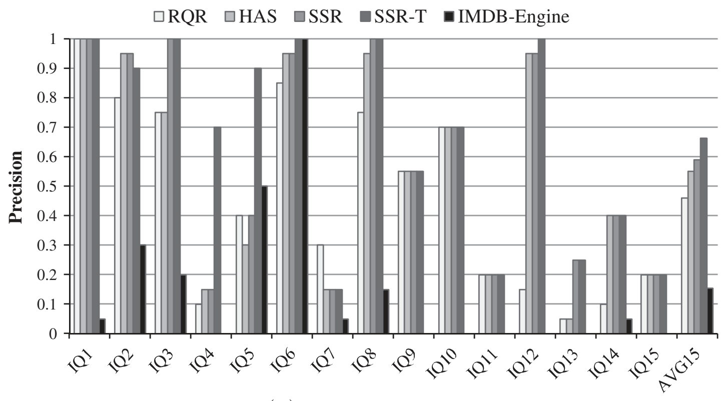


(c) Top-10 F-Score

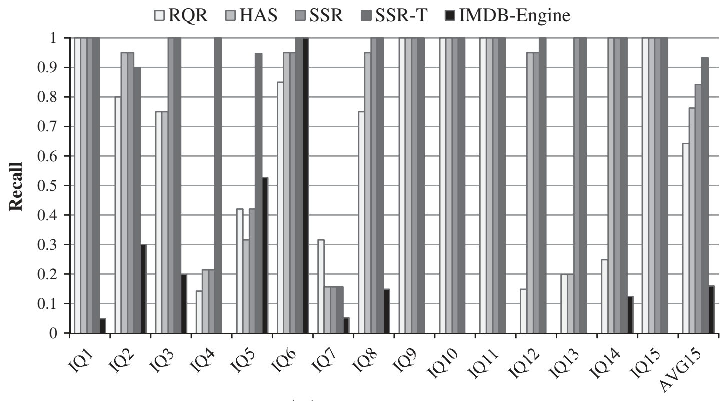
Fig. 10. The accuracy for *top*-10 IMDB query results.

8. Conclusions

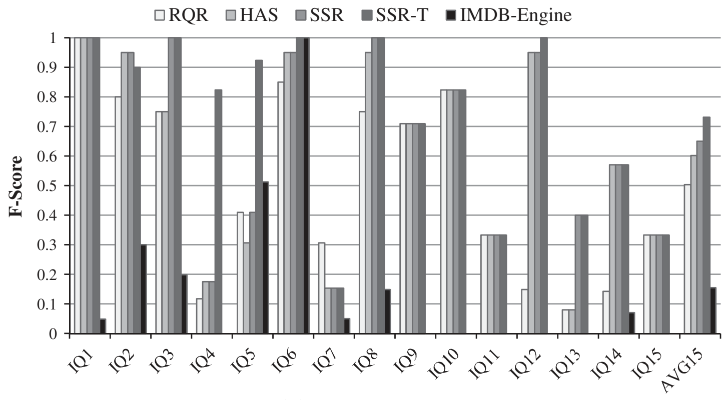
In this paper, we studied effective ranking and semantic search techniques to provide accurate results by using an ontology. The goal of our semantic search is to retrieve *top-k* results which are fairly relevant to many query keywords which significantly contribute to the relevance. To do this, we first devised a weighting measure to determine the relative importance of a semantic path to determine the relevance of resources. Then, based on this measure, we proposed a new ranking method which reflects the weights of semantic paths, the coverage of keywords, and the discriminating power of keywords. In addition, we pruned superfluous semantic paths based on the length and weight of the semantic paths since the pruned semantic paths incur massive traversals in the ontology and even deteriorate the accuracy of the search. Moreover, we avoided the traversal of a large sized instance graph during a query time through the *top-k* answering utilizing Threshold Algorithm



(a) Top-20 Precision



(b) Top-20 Recall



(c) Top-20 F-Score

Fig. 11. The accuracy for *top*-20 IMDB query results.

based on the keyword index. Through the experiments using real data sets, we observed that our ranking method provided more accurate search results compared to existing ranking methods. In addition, our pruning algorithm effectively reduced the search space and even improved the accuracy. Finally, we observed the efficiency of our *top-k* semantic search processing.

Acknowledgements

We would like to thank the editor and anonymous reviewers. This work was supported by the National Research Foundation of Korea grant funded by the Korean government (MSIP) (No. NRF-2009-0081365).

Appendix A

See Figs. 8–11.

References

Alani, H., Brewster, C., & Shadbolt, N. (2006). Ranking ontologies with AKTiveRank. In *Proceedings of the 5th International Semantic Web conference. Lecture notes in computer science* (Vol. 4273, pp. 1–15). Springer.

Anyanwu, K., Maduko, A., & Sheth, A. (2005). SemRank: Ranking complex relationship search results on the Semantic Web. In *Proceedings of the 14th international conference on World Wide Web* (pp. 117–127). ACM (May).

Anyanwu, K., & Sheth, A. (2003). ρ-Queries: Enabling ouerying for semantic associations on the Semantic Web. In *Proceedings of the 12th international conference on World Wide Web* (pp. 690–699). ACM (May).

Castells, P., Fernandez, M., & Vallet, D. (2007). An adpatation of the vector-space model for ontology-based information retrieval. *IEEE Transactions on Knowledge and Data Engineering, 19*(2), 261–272 (February).

Ceravolo, P., & Damiani, E. (2007). Bottom-up extraction and trust-based refinement of ontology metadata. *IEEE Transactions on Knowledge and Data Engineering, 19*(2), 149–163 (February).

Cover, T. M. (1991). *Elements of information theory*. New York, NY, USA: John Wiley and Sons Inc.,.

Fagin, R., Lotem, A., & Naor, M. (2003). Optimal aggregation algorithms for middleware. *Journal of Computer and System Sciences, 66*(4), 614–656 (June).

Guo, L., Shao, F., Botev, C., & Shanmugasundaram, J. (2003). XRANK: Ranked keyword search over XML documents. In *Proceedings of the ACM SIGMOD international conference on management of data* (pp. 16–27). ACM (June).

Hristidis, V., & Papakonstantinou, Y. (2002). DISCOVER: Keyword search in relational databases. In *Proceedings of 28th international conference on very large data bases* (pp. 670–681). ACM (August).

Hull, D. (1993). Using statistical testing in the evaluation of retrieval experiments. In *Proceedings of the 16th annual international ACM SIGIR conference on research and development in information retrieval. SIGIR '93* (pp. 329–338). New York, NY, USA: ACM. <http://dx.doi.org/10.1145/160688.160758>.

Kiryakov, A., Popov, B., Ognyanoff, D., Manov, D., Kirilov, A., Goranov, M., et al (2003). Semantic annotation, indexing, and retrieval. In *Proceedings of the 2nd international Semantic Web conference. Lecture notes in computer science* (Vol. 2870, pp. 484–499). Springer.

Li, Y., Wang, Y., & Huang, X. (2007). A relation-based search engine in Semantic Web. *IEEE Transactions on Knowledge and Data Engineering, 19*(2), 273–282 (February).

Liu, F., Yu, C., Meng, W., & Chowdhury, A. (2006). Effective keyword search in relational databases. In *Proceedings of the ACM SIGMOD international conference on management of data* (pp. 563–574). ACM (June).

Luo, Y., Lin, X., Wang, W., & Zhou, X. (2007). SPARK: Top-k keyword query in relational databases. In *Proceedings of the ACM SIGMOD international conference on management of data* (pp. 115–126). ACM (June).

Manning, C. D., Raghavan, P., & Schutze, H. (2008). *Introduction to information retrieval*. Cambridge University Press.

Resnik, P. (1999). Semantic semilarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research, 11*, 95–130.

Rocha, C., Schwabe, D., & de Aragao, M. P. (2004). A hybrid approach for searching in the Semantic Web. In *Proceedings of the 13th international conference on World Wide Web* (pp. 374–383). ACM (May).

Salton, G., Fox, E. A., & Wu, H. (1983). Extended Boolean information retrieval. *Communications of ACM, 26*(12), 1022–1036 (December).

Salton, G., & McGil, M. J. (1986). *Introduction to mordern information retrieval*. New York, NY, USA: McGraw-Hill Inc.,.

Stojanovic, N., Studer, R., & Stojanovic, K. (2003). An approach for the ranking of query results in the Semantic Web. In *Proceedings of the 2nd international semantic web conference. Lecture notes in computer science* (Vol. 2870, pp. 500–516). Springer (October).

Suchanek, F. M., Kasneci, G., & Weikum, G. (2007). YAGO: A core of semantic knowledge unifying WordNet and Wikipedia. In *Proceedings of the 16th international conference on World Wide Web* (pp. 697–706). ACM (May).

Theobald, M., Schenkel, R., & Weikum, G. (2005). An efficient and versatile query engine for TopX search. In *Proceedings of 31th international conference on very large data bases* (pp. 625–636). ACM (August).

Wu, G., Li, J., Feng, L., & Wang, K. (2008). Identifying potentially important concepts and relations in an ontology. In *Proceedings of the 7th international semantic web conference. Lecture notes in computer science* (Vol. 5318, pp. 33–49). Springer.

Zhou, J., Chen, Z., Tang, X., Bao, Z., & Ling, T. W. (2012). Fast result enumeration for keyword queries on xml data. *JCSE, 6*(2), 127–140.