An Enhanced Model for Searching in Semantic Portals

Lei Zhang, Yong Yu,
Jian Zhou, ChenXi Lin
APEX Data and Knowledge Management Lab
Dept. of Computer Science and Engineering
Shanghai JiaoTong University
Shanghai, 200030, China
{zhanglei,yyu,priest,linchenxi}@apex.sjtu.edu.cn

Yin Yang*
Dept. of Computer Science
HongKong University of Science and Technology
Clear Water Bay, Hong Kong, China
yini@cs.ust.hk

ABSTRACT

Semantic Portal is the next generation of web portals that are powered by Semantic Web technologies for improved information sharing and exchange for a community of users. Current methods of searching in Semantic Portals are limited to keyword-based search using information retrieval (IR) techniques, ontology-based formal query and reasoning, or a simple combination of the two. In this paper, we propose an enhanced model that tightly integrates IR with formal query and reasoning to fully utilize both textual and semantic information for searching in Semantic Portals. The model extends the search capabilities of existing methods and can answer more complex search requests. The ideas in a fuzzy description logic (DL) IR model and a formal DL query method are employed and combined in our model. Based on the model, a semantic search service is implemented and evaluated. The evaluation shows very large improvements over existing methods.

Categories and Subject Descriptors

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

General Terms

Algorithms, Experimentation

Keywords

semantic search, semantic portal, fuzzy description logic, fuzzy reasoning, information retrieval

1. INTRODUCTION

Semantic Portal [24] is the next generation of web portals that are powered by Semantic Web technologies for improved information sharing and exchange for a community of users. It has an immediate and important bussiness application — knowledge management in enterprise intranet portals. In recent years, many research and development efforts were devoted to this area. An initial survey on it can be found in [12]. Briefly speaking, information in a semantic portal is organized by a domain ontology and stored in a portal knowledge base (KB). The portal is then dynamically generated by a runtime system from the portal KB for ontology-based browsing, searching and editing. Semantic Portal improves the way web portals are generated and maintained. It also better facilitates the dissemination and sharing of the information in the portal for both human and machine consumption.

This paper focuses on how search is performed in semantic portals. Current methods include: (1) IR-based keyword search (2) ontology-based formal query and reasoning (3) simple combination of the above two methods. Method (2) is a unique feature of semantic portals because traditional web portals have no semantic information to use. However, the drawback of method (2) is that it can hardly exploit textual information in the portal. It is estimated that 80% of enterprise information is in textual form, and we believe that textual information still prevails in semantic portals, especially in enterprise knowledge portals. Thus a method must be used to fully exploit these textual information. This calls for an effective combination of methods (1) and (2). Otherwise, the portal users will be left with two separate search methods at two extremes. Method (2) uses only semantic information and either returns all correct answers for a formal query or none at all. Method (1), on the contrary, searches only textual information and typically returns a lot of answers with many irrelevant ones. The challenge is then how to effectively combine them to obtain a smooth model between the two extremes and better satisfy users' information needs.

In the Semantic Portal area, current solutions to the challenge, i.e. method (3), are quite basic. In OntoWeb portal[1], when an ontology-based query has no results to return, it is then automatically converted to a pure keyword search. In the commercial product Mondeca ITM [2], users can specify keywords that the searched item must contain and concepts/relations they must belong/relate to [12]. Analysis of other related work (e.g. work in the semantic search and retrieval area) in Section 6 shows that they are also limited in one way or another. These existing methods can not effectively express and answer search requests like *"find documents talking about future market that are written by some senior staff related to semantic technology"*. In this search

*The author's work is done in Shanghai JiaoTong University.

WWW 2005, May 10-14, 2005, Chiba, Japan.
ACM 1-59593-046-9/05/0005.

[1] www.ontoweb.org

[2] http://www.mondeca.com/technologie.htm

request, *"documents talking about future market"* may be retrieved using IR method. The retristriction *"written by some senior staff"* should be checked against the portal knowledge base by reasoning on the document authors and on who are qualified to be senior staffs (e.g. only managers and employees in some key positions are qualified). Whether the inferred senior staffs are somehow *"related to semantic technology"* probably again involves a keyword search on their properties in the portal KB using "semantic technology" as keywords. Therefore, we need a very tight integration of IR and formal query and reasoning to answer this search request. Clearly, the ability to answer this kind of search requests should be a key feature of semantic portals and is very critical in applications like business intelligence.

In this paper, we propose a model that enables such tight integration between IR and formal query and reasoning to fully utilize both textual and semantic information in semantic portals. The capability of the model ranges from simple IR and purely formal query and reasoning to complex integration of both. The ideas in a fuzzy DL based IR model [15] and a formal DL query method [11] are employed and combined in our model. We implemented a semantic search service based on the model and carried out an evaluation. Compared to existing search methods, this model shows very large improvements in the evaluation.

The rest of the paper is organized as follows. The model is first presented in detail in Section 2 and then its capabilities, properties and limitations are discussed in Section 3. After that, we describe the implementation of the model in Section 4 and show the evaluation in Section 5. Related work is discussed in Section 6 and Section 7 concludes the paper.

2. THE MODEL

Our model largely draws upon and is also inspired by the body of work on IR models based on Description Logics (DL) [15, 16, 17, 20]. In DL-based IR models, documents and queries are modeled as DL individuals and concepts respectively. The content, structure, layout, thesaural information of all the documents are also described in DL and form a document knowledge base Σ. Whether a document d is relevant to a query Q is modeled as whether $\Sigma \models d : Q$. The IR problem is then reduced to the DL *instance retrieval* problem which can be answered by a DL reasoning engine. On the other hand, [11] showed that many formal DL conjunctive queries can be "rolled up" to a single concept expression and reduced to the DL instance retrieval problem too. The difference is that the IR model can not query non-document objects using semantic information, while the formal query method can not use IR techniques to exploit textual information.

This leads to our intuition that if we extend the idea $\Sigma \models d : Q$ to one that retrieves both documents and non-document objects and can also integrate them together, we can obtain an enhanced model that utilizes both textual and semantic information. Following this intuition and also in order to make the paper self-contained, we first introduce the fuzzy description logic fuzzy-$\mathcal{ALC}$ [26] in section 2.1. We then present the basic model, the extended model and the complete model in section 2.2, 2.3 and 2.4 respectively.

2.1 Fuzzy-$\mathcal{ALC}$

IR methods represents the degree of relevance of a document w.r.t a query using the so-called RSV (Retrieval Status Value) that is usually normalized in $[0, 1]$. On the contrary, formal query and reasoning mechanisms are usually based on a binary judgement (i.e., either 0 or 1). Fuzzy description logic introduces uncertainty (i.e., non-binary set membership degree between 0 and 1) into DL. It hence can serve as a good glue to combine IR and formal methods. Here we briefly summarize the work done by Umberto Straccia on Fuzzy-$\mathcal{ALC}$ in [26]. We assume the readers are familiar with the basics of description logics.

The syntax of concept descriptions and TBox axioms of fuzzy-$\mathcal{ALC}$ is the same as the standard $\mathcal{ALC}$ DL. The concept description syntax is defined as

$$C, D \rightarrow \top \mid \bot \mid A \mid C \sqcap D \mid C \sqcup D \mid \neg C \mid \forall R.C \mid \exists R.C$$

TBox axioms are either of the form $A \sqsubseteq C$ or $A = C$. What Fuzzy-$\mathcal{ALC}$ differs is in the ABox where *fuzzy assertions* can be expressed in the form of $\langle \alpha \leq m \rangle$ or $\langle \alpha \geq n \rangle$. Here, α is a standard $\mathcal{ALC}$ ABox assertion $a : C$ or $(a, b) : R$ and $m \in [0, 1), n \in (0, 1]$ represents the degree of fuzziness of α. For example, the fuzzy assertion $\langle Tom : Tall \geq 0.8 \rangle$ says that Tom is a tall person with a degree greater than 0.8. Note that this syntax also allows us to assert $\langle \alpha = n \rangle$ because it is equivalent to asserting both $\langle \alpha \leq n \rangle$ and $\langle \alpha \geq n \rangle$.

The semantics of Fuzzy-$\mathcal{ALC}$ is defined on fuzzy interpretations. A *fuzzy interpretation* $\mathcal{I}$ is a pair $(\triangle^{\mathcal{I}}, \cdot^{\mathcal{I}})$ where $\triangle^{\mathcal{I}}$ is a domain and $\cdot^{\mathcal{I}}$ is a function mapping a concept C into a function $C^{\mathcal{I}} : \triangle^{\mathcal{I}} \rightarrow [0, 1]$ and a role R into a function $R^{\mathcal{I}} : \triangle^{\mathcal{I}} \times \triangle^{\mathcal{I}} \rightarrow [0, 1]$. That is, $\mathcal{I}$ interprets concepts and roles as fuzzy set *membership degree functions*. In contrast, in standard DL, concepts and roles are interpreted as crisp sets. A fuzzy interpretation $\mathcal{I}$ must satisfy the following equations (for any $d \in \triangle^{\mathcal{I}}$):

$$\begin{aligned}
\top^{\mathcal{I}}(d) &= 1 \\
\bot^{\mathcal{I}}(d) &= 0 \\
(C \sqcap D)^{\mathcal{I}}(d) &= \min\left\{C^{\mathcal{I}}(d), D^{\mathcal{I}}(d)\right\} \\
(C \sqcup D)^{\mathcal{I}}(d) &= \max\left\{C^{\mathcal{I}}(d), D^{\mathcal{I}}(d)\right\} \\
(\neg C)^{\mathcal{I}}(d) &= 1 - C^{\mathcal{I}}(d) \\
(\forall R.C)^{\mathcal{I}}(d) &= \inf_{d' \in \triangle^{\mathcal{I}}} \left\{\max\left\{1 - R^{\mathcal{I}}(d, d'), C^{\mathcal{I}}(d')\right\}\right\} \\
(\exists R.C)^{\mathcal{I}}(d) &= \sup_{d' \in \triangle^{\mathcal{I}}} \left\{\min\left\{R^{\mathcal{I}}(d, d'), C^{\mathcal{I}}(d')\right\}\right\}
\end{aligned}$$

These equations are the standard fuzzy logic interpretation of conjunction, disjunction, negation and qualification, respectively. Note that the semantics of $\forall R.C$ is the result of viewing $\forall R.C$ as the first order formula $\forall d' \neg R(d, d') \vee C(d')$, where the universal quantification $\forall$ is viewed as a conjunction over all $d' \in \triangle^{\mathcal{I}}$. The interpretation of $\exists R.C$ is similar. We say that $\mathcal{I}$ *satisfies*

$$\begin{aligned}
A \sqsubseteq C \quad &\text{iff} \quad \forall d \in \triangle^{\mathcal{I}}, A^{\mathcal{I}}(d) \leq C^{\mathcal{I}}(d) \\
A = C \quad &\text{iff} \quad \forall d \in \triangle^{\mathcal{I}}, A^{\mathcal{I}}(d) = C^{\mathcal{I}}(d) \\
\langle a : C \geq / \leq n \rangle \quad &\text{iff} \quad C^{\mathcal{I}}(a^{\mathcal{I}}) \geq / \leq n \\
\langle (a, b) : R \geq / \leq n \rangle \quad &\text{iff} \quad R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}}) \geq / \leq n
\end{aligned}$$

A fuzzy interpretation $\mathcal{I}$ is a *model* of a fuzzy-$\mathcal{ALC}$ knowledge base Σ iff it satisfies every TBox axiom and ABox fuzzy assertions of Σ. Finally, the *fuzzy entailment* $\Sigma \models \langle \alpha \geq n \rangle$ holds iff every model of Σ also satisfies $\langle \alpha \geq n \rangle$.

Given a fuzzy KB Σ and an ABox assertion α, it is of interest to compute α's best lower and upper fuzzy degree bounds. Formally, we define the *greatest lower bound* of α w.r.t Σ, denoted $glb(\Sigma, \alpha)$, to be $\sup\{n \mid \Sigma \models \langle \alpha \geq n \rangle\}$. Similarly, the *least upper bound* of α w.r.t Σ, $lub(\Sigma, \alpha)$, is defined to be $\inf\{n \mid \Sigma \models \langle \alpha \leq n \rangle\}$. Determining the *lub*

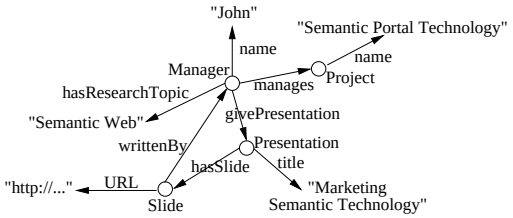

Figure 1: An example graph fragment

and the *glb* is called the *Best Truth-Value Bound* (BTVB) problem. [26] shows that the BTVB problem can be reduced to the entailment problem, and in turn to the satisfiability problem which can then be solved using a tableau algorithm for fuzzy-$\mathcal{ALC}$. This result will be used in this paper.

2.2 The Basic Model

In a semantic portal, information is organized by a domain ontology and stored in a knowledge base. The ABox of the knowledge base can be visualized as a labeled graph. The nodes are the individuals and the edges connect individuals with role relationships. Labels on the nodes represent the asserted concepts the individual belongs to. Labels on the edges are the role names. Fig. 1 shows an example graph fragment. In this paper, instead of directly talking about the ABox of the portal KB, we usually talk about the graph. The term *nodes* and *individuals* are also used alternatively in this paper. Given the portal KB, we can directly apply the formal query method [11] for searching in the portal. A query is modeled as a DL concept Q. Answers to the query are individuals of the concept Q and can be retrieved using the DL instance retrieval algorithm. For example, we can search for all senior staff that give at least a presentation using the query:

$$\texttt{SeniorStaff} \sqcap \exists \texttt{givePresentation}.\texttt{Presentation}$$

In the Fig. 1 example, the node representing John may be deduced as an answer because John is a manager and may be inferred to be a senior staff according to the domain ontology and John does give a presentation. The problem with the formal query method is that it does not incorporate IR to retrieve document nodes in the graph. For example, we can not use IR in the formal query method to answer the query Q_{IR}: *find presentation slides that talk about future market.*

We now add IR capability to the formal query method to obtain our basic model. The intuition is that we can still view the IR query Q_{IR} as a concept. Documents that are IR-relevant to the query can be seen as the individuals of the concept with a fuzzy degree. The degree value is between $[0, 1]$ and is determined by the IR *retrieval status value* (RSV). In other words, Q_{IR} is modeled as a *fuzzy concept* in a fuzzy DL. For the above example, we can express Q_{IR} as

$$Q_{IR} : \texttt{Slide} \sqcap D_q$$

where D_q is the fuzzy concept denoting the set of documents that are relevant to the IR query q: *about future market.* For any presentation slide s, the degree of s belonging to the `Slide` concept, denoted $s : \texttt{Slide}$, is 1. The degree $s : D_q$ is the RSV of the text content of slide s w.r.t the IR query q — $\text{RSV}(t(s), q)$ where $t(s)$ denotes the text content of s. According to the fuzzy interpretation of conjunction introduced in section 2.1, the degree $s : Q_{IR} = \min\{1, \text{RSV}(t(s), q)\} = \text{RSV}(t(s), q)$. On the contrary, for any document d that is not a presentation slide, the degree $d : \texttt{Slide}$ is 0 and hence the degree $d : Q_{IR} = \min\{0, \text{RSV}(t(d), q)\} = 0$. Thus, the results of the query Q_{IR} are those document nodes d that have a degree $d : Q_{IR} > 0$. In addition, the results can be ranked according to the degree in descending order. In Fig. 1, the bottom node may be returned as one of the good answers of the Q_{IR} query because it is a presentation slide and it talks about marketing. Once IR is incorporated in this way, fuzzy IR concepts can be placed in any position of a formal DL query that a concept can appear. Consequently, more interesting and complex queries than Q_{IR} can be expressed. For example, the following query

$$\texttt{SeniorStaff} \sqcap \exists \texttt{givePresentation}.(\exists \texttt{hasSlide}.D_q)$$

finds all senior staff giving a presentation that has a slide talking about future market. This query involves both reasoning (on senior staff) and IR (on the slide content).

The basic model can be described more formally as follows. We assume the portal KB Σ is in standard $\mathcal{ALC}$ DL. Because the standard $\mathcal{ALC}$ is a special case of the fuzzy-$\mathcal{ALC}$ with the fuzzy degree being 1, we can easily convert Σ to an equivalent fuzzy-$\mathcal{ALC}$ KB Σ_F by simply transforming the Σ ABox assertions α to $\langle \alpha \geq 1 \rangle$. In addition, in the Σ TBox, it is reasonable to assume that there is a concept, denoted `Doc`, representing all the document nodes in the KB.[3] The set of document nodes is then defined as $D_\Sigma = \{d \,|\, \Sigma \models d : \texttt{Doc}\}$. In the model, the user's information need in searching the portal is modeled as a fuzzy concept Q. Concept names in Q either are defined in the TBox of Σ_F or are new fuzzy concept names representing the IR queries in Q (e.g. the D_q in the above metioned examples). Let's denote the set of new IR fuzzy concept names in Q as $\mathbb{C}_Q^{IR}$. The KB Σ_F is then enriched with a set of new fuzzy assertions about the relevances of document nodes w.r.t IR queries in Q. We define the enriched Σ_F KB as Σ'_F:

$$\Sigma'_F = \Sigma_F \cup \left\{ \langle i : D_q = \text{RSV}(t(i), q) \rangle \;\middle|\; \forall i \in D_\Sigma, \forall D_q \in \mathbb{C}_Q^{IR} \right\}$$

The function RSV calculates the IR relevance of a text content w.r.t an IR query and returns a degree in [0,1]. Note that RSV is applied only on individuals $i \in D_\Sigma$. That is, IR method is only used on documents. Another point to note is that, if Q is a pure formal query that contains no IR queries, $\mathbb{C}_Q^{IR}$ is $\emptyset$ and $\Sigma'_F = \Sigma_F \cup \emptyset = \Sigma_F$. Based on Σ'_F, the answer set of Q is then defined as the set of nodes whose greatest lower bound degree of being individuals of Q is greater than 0: $\{i \,|\, glb\,(\Sigma'_F, i : Q) > 0\}$[4]. Recall that the *glb* values can be calculated by fuzzy-$\mathcal{ALC}$ algorithms. The answer set can then be ranked according to the *glb* values.

One limitation of the basic model is that we restrict the portal KB to be a $\mathcal{ALC}$ DL KB. The reason is that our model relies on the fuzzy-$\mathcal{ALC}$ formalism. However, this does not impose much restriction on the applicability of our model. First, ontologies in real applications usually do not use very expressive DL constructs. Analysis [29] shows that ontologies in the DAML ontology library seldom use very expressive DL constructs. Second, some existing semantic portal

[3]The concept `Doc` may be definded and added to the TBox if it is not already there.

[4]The use of *glb* values instead of *lub* values is consistent with [15] and is also more rational.

systems, such as OntoWeb, are based on F-logic, which corresponds to $\mathcal{ALUN}$, a DL relates to $\mathcal{ALC}$. Third, since $\mathcal{ALC}$ is a significant and expressive representative of the various DLs, the principles discussed in this paper are possible to be adapted to KBs based on other DLs. For a KB based on a more expressive DL, we can even partially circumvent the problem by extracting the $\mathcal{ALC}$ part of the KB and then applying our model. Another limitation of the basic model is that IR method is only used for searching documents. In the next subsection, we remove the latter limitation by enabling the use of IR method also for searching non-document individuals in the portal KB.

2.3 Extending The Basic Model

Most existing semantic portal systems support the use of keyword-based search for finding non-document individuals in the portal KB. Any individual that has a string property that matches the search keywords will be returned as a result. However, this is not supported in our basic model.

The problem stems from the fact that in the basic model non-document nodes have no text representations and thus no IR RSV values to be added to the Σ'_F KB. The problem can be solved by endowing every non-document node with a text representation so that they can also be indexed and searched by IR methods and have RSV values. First, we define the *simple text representation* of a node o, $str(o)$, as a string that people usually use to talk about the node, e.g. the name of a people, the title of a presentation, etc. Usually, $str(o)$ equals to a string property of the node.

The *text representation*, $tr(o)$, of a node can then be constructed from the simple text representaions of its surrounding nodes. The intuition is that the properties and relations of a node implicitly define, and thus represent the node. For a document node o, $tr(o)$ also needs include its text content. However, the specific method used for building the text representation varies and depends on the IR method used. We list three possible methods as examples for the model.

Simple Concatenation Method $tr(o)$ is a piece of text that is the result of a concatenation of all the string properties of o and all the $str(p)$ of its surrounding nodes p that has an edge directly connected to o. If o is a document node, its text content is also concatenated.

Weighted Term Method For all text terms in the $str(p)$ of the surrounding nodes p of o, a weight is assigned based on the distance between p and o. For other text terms in the string properties of o, or in the text content of o if o is a document node, a weight is assigned as if the distance is 1. $tr(o)$ is then composed as a weighted term vector.

PLBR Method PLBR [6] is an IR model based on propositional logic. Using this model, $tr(o)$ is a DNF formula $dc_1 \vee dc_2 \vee \ldots \vee dc_n$ where each dc_i is a conjunction of all text terms in one string property of o, or in one $str(p)$ of a directly connected node p of o, or in the text content of o if o is a document node.

Let's take the node representing John in Fig. 1 as an example. With the simple concatenation method, $tr(o)$ is the string: "John Semantic Web Semantic Portal Technology Marketing Semantic Technology". With the weighted term method, $tr(o)$ may be a vector like <(John, 0.5), (Semantic, 0.6), (Technology, 0.3), (Web, 0.5), (Portal, 0.15), (Marketing, 0.15)>. These two methods are suitable for the classical vector space IR model. With the PLBR method, $tr(o)$ is the DNF formula: John $\vee$ (Semantic $\wedge$ Web) $\vee$ (Semantic $\wedge$ Portal $\wedge$ Technology) $\vee$ (Marketing $\wedge$ Semantic $\wedge$ Technology). PLBR method captures the different properties/relations of an individual in different conjunctive clauses in a DNF formula. This makes it suitable for representing a node in the graph with different connections.

Once we have a text representation for every node in the portal KB Σ, we can extend the original RSV function to also cover non-document nodes. We define the function RSV′:

$$\forall o \in \Sigma : \text{RSV}'(o, q) = \text{RSV}(tr(o), q)$$

The RSV′ function abstracts both the method used to construct the text representation $tr(o)$ and the IR method by which the relevance between $tr(o)$ and q is calculated. The premise is that the constructed $tr(o)$ must be suitable for the IR method used to calculate the RSV. For example, if $tr(o)$ is constructed using the PLBR method, then the PLBR retrieval model must be used to calculate RSV. Another observation is that, because every node now has a text representation, we actually obtained a *virtual document collection* VDC_Σ from the portal KB Σ:

$$VDC_\Sigma = \{\, tr(o) \,|\, \forall o \in \Sigma \,\}$$

Many existing IR techniques on document collections, such as inverse document frequency and latent semantic indexing [5], can be readily applied on the collection VDC_Σ to help better calculate the RSV values. With this newly defined RSV′ function, we can now give the complete model.

2.4 The Complete Model

The complete model is almost the same as the basic model except for the definition of Σ'_F. In the basic model, we have no RSV values for non-document nodes to add to Σ'_F. With the extension introduced in the previous subsection, both non-document nodes and document nodes have RSV values defined by the RSV′ function. So now

$$\Sigma'_F = \Sigma_F \cup \left\{ \langle i : D_q = \text{RSV}'(i, q) \rangle \;\middle|\; \forall i \in \Sigma_F, \forall D_q \in \mathbb{C}_Q^{IR} \right\}$$

Compared with the basic model, the variable i is no longer limited to the document nodes alone, and can now range over all the individuals in the portal KB. All other things of the complete model are exactly the same as the basic model. The result set is still $\{i \,|\, glb\,(\Sigma'_F, i : Q) > 0\}$ and the glb values can be used for ranking the results.

3. DISCUSSION

3.1 Capabilities of The Model

After presenting the model, we now explain the wide range of search capabilities of the model and show that it is a smooth model between the pure IR method and the pure formal query method, and it indeed fully utilizes both textual and semantic information in semantic portals.

First, if the query Q of the search request is a single IR fuzzy concept, Q is then a pure IR query. For example, the query

$$Q1 :\ D_q$$

is a pure IR query q. According to the definition of Σ'_F, the only fuzzy assertions added is $\langle i : D_q = \mathrm{RSV}'(tr(i), q)\rangle$. These are also the only fuzzy assertions related to the *glb* values in the result set $\{i \,|\, glb\,(\Sigma'_F, i : D_q) > 0\}$. It follows that the *glb* values equal to the RSV′ values, which means that the ranked result set is the same as the ranked IR relevance set. Therefore, the model degenerates to a pure IR method. Note that even in this simple IR case, we are actually searching for both documents and non-document individuals. For non-document individuals, we are relying on their text representations.

Second, if the query Q of the search request contains no IR fuzzy concept, Q then becomes a pure formal query. The query

$$Q2: \texttt{SeniorStaff} \sqcap \exists \texttt{givePresentation.Presentation}$$

is such an example. It finds senior staff that gives at least one presentation. In this case, as we have shown in section 2.2, $\mathbb{C}_Q^{IR} = \emptyset$ and therefore $\Sigma'_F = \Sigma_F \cup \emptyset = \Sigma_F$. That is, the Σ'_F KB is the same as the fuzzy-$\mathcal{ALC}$ KB of the original portal KB Σ. It can be easily proved from [26] that under this situation the result set $\{i \,|\, glb\,(\Sigma'_F, i : Q) > 0\}$ is the same as $\{i \,|\, \Sigma \models i : Q\}$. In other words, the model now degenerates to the basic form of the formal query and reasoning method in [11]. About the capability in this mode, [11] made a good discussion.

Between the above two extremes of search methods, the model can support, from simple to complex, integration of IR and formal query and reasoning. A simple integration example is the query

$$Q3: \texttt{SeniorStaff} \sqcap D_q$$

where D_q is a fuzzy concept of IR query q: *about semantic technology*. The query finds all senior staff related to semantic technology. It involves both reasoning on who are senior staff and IR on the text representations of the inferred senior staff. It is a typical query supported by the simple combination method used by most existing semantic portal systems. In the simple combination method, a query is a conjunction of a pure formal sub-query and a pure IR sub-query. $Q3$ is exactly such a query. Our model now degenerates to the simple combination method to support this query. However, the simple combination method does not support more interesting and deeper integration of IR and formal query and reasoning. For example, the query

$$Q4: \texttt{Doc} \sqcap \exists \texttt{writtenBy.}(\texttt{SeniorStaff} \sqcap D_q)$$

searches for documents that are written by some senior staff related to semantic technology. It uses IR to restrict the `SeniorStaff` concept in a $\exists R.C$ construct of a formal query. An even more complex query is

$$Q5: (\texttt{Doc} \sqcap D_p) \sqcap \exists \texttt{writtenBy.}(\texttt{SeniorStaff} \sqcap D_q)$$

where D_p is a fuzzy concept of IR query p: *about future market*. The query searches for documents talking about future market that are written by some senior staff related to semantic technology. $Q4$ and $Q5$ are two typical examples of complex but very valuable queries in real applications. Answering them requires very deep integration of implicit semantics in texts/documents and explicit semantics in knowledge markups. They exceed the search capability of the simple combination method. However, they can be naturally expressed and supported in our model.

Through the $Q1$ to $Q5$ query examples, it is clear that the model can integrate both IR and formal query and reasoning to utilize both textual and semantic information in the portal for answering search requests. The model is indeed a smooth model between the two extremes of search methods. In addition, being able to answer complex queries like $Q4$ and $Q5$, which can hardly be answered by existing methods, shows that the model indeed extends the search capabilities of existing methods.

3.2 Properties of The Model

Besides the wide range of search capabilities, this model also enables the mutual assistance of IR and formal query and reasoning in real applications to deliver better search results.

On the one hand, IR can help formal query and reasoning in this model. Although the latter one is good at utilizing explicit semantics, it can hardly use implicit semantics buried in large amount of texts. IR method, such as LSI [5], can discover and use implicit semantics such as synonymous words and similar documents to deliver better search results. IR method can also help the formal method when there is a lack of semantic information in the portal. For example, the following query intends to find papers on IR topics

$$\texttt{Paper} \sqcap \exists \texttt{hasTopic.IRTopics}$$

However, if papers are very lazily or inadequately marked up in the portal on its topics, the query won't return many good results. In this case, we can use IR to relax the query to

$$\texttt{Paper} \sqcap (D_q \sqcup \exists \texttt{hasTopic.IRTopics})$$

where D_q is a query like "about information retrieval, document, precision, recall". For this relaxed query, if a paper is explicitly marked up as an IR paper, it will surely be returned and ranked high in the result. More interestingly, papers not explictly marked up so but actually about IR topics may also appear in the result list along with those explicitly marked ones.

On the other hand, formal query and reasoning also can help IR in the model. It can improve IR precision because logic conditions can greatly reduce the scope of retrieval. It can also improve recall because reasoning can discover information that is not explicitly stated. For example, people usually won't say he/she is a senior staff on his/her homepage and may not be found by IR queries using "senior staff" as keywords. Formal reasoning, however, can discover this unstated information and consequently improve the recall.

3.3 Limitations of The Model

Despite the various merits disccused above about the model, we are clearly aware that the model has some limitations. First, the fuzzy-$\mathcal{ALC}$ formalism used in the model has its own limitations when applied to information retrieval and search. We've discussed the $\mathcal{ALC}$ limitation in section 2.2. Actually, in general, DL-based IR models have various pros and cons as discussed in the literature [15, 16, 17, 20]. Regarding to fuzzy logic, it also has well-known problems when applied to IR. For example, for a query $A \sqcap B$, if we have an individual a belonging to A but not B, and another individual c not belonging to either A or B, the fuzzy logic approach would treat a and c indifferently as not being relevant to the query at all. However, a more natural view is that a is more relevant to the query than c.

The second problem lies in the end user's difficulty in converting the information need in a search to a well formulated query of this model. In many situations, users are more adapted to use just keywords. Although this is more about a human computer interface issue and not directly related to the model itself, it inevitably affects the applicability of the model. Recent work on ontology-based visual query formulation such as [3, 2] bring initial solutions to the problem.

Third, our model relies on a fuzzy-$\mathcal{ALC}$ reasoning engine to calculate the *glb* values for all individuals in the portal KB. To be used in large applications with large amounts of individuals, the reasoning engine must be able to scale up. Currently, the only fuzzy-$\mathcal{ALC}$ engine we know of is alc-F [28]. It is a very naive implementation and runs very slowly in our model implementation and evaluation. Nevertheless, there is still plenty of room for performance improvements. Internally, the engine can adapt and reuse many proven optimization techniques for DL reasoning. One way to achieve this is to transform fuzzy DL reasoning to crisp DL reasoning [27]. Externally, indexing and caching techniques can be used to greatly reduce the number of reasoning tasks. We discuss the semantic index used in our current implementation in section 4.1.

4. IMPLEMENTATION

We've developed a semantic search service implementing our model in the SPortS [13] semantic portal system. Given an OWL-DL portal KB and a site specification, SPortS can dynamically generate a web portal for ontology-based information browsing and editing. Different from many other semantic portal systems, SPortS has the ability to integrate semantic web services into it to provide more up-to-date information and richer functionalities. Utilizing this feature, we implemented the semantic search function as a semantic web service to be integrated into the portal. Presently, we do not have a GUI to support end user query formulation and input. The service can only be accessed programmatically.

The semantic search service accepts a query concept Q as input and returns a list of URIs of individuals in the portal KB as the answer. Because Q is an $\mathcal{ALC}$ concept, it can be easily encoded using the OWL RDF/XML syntax. The following is a snippet of the encoded query $\texttt{Doc} \sqcap D_q$:

```
<owl:Class rdf:ID="D_q">
  <IRQuery xml:lang="en">future market</IRQuery>
</owl:Class>
<owl:Class rdf:ID="Q">
  <owl:equivalentClass>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <owl:Class rdf:about="&portal;Doc"/>
        <owl:Class rdf:about="#D_q"/>
      </owl:intersectionOf>
    </owl:Class>
  </owl:equivalentClass>
</owl:Class>
```

The `IRQuery` property in the code is an annotation property of the fuzzy IR concept D_q. After the query is processed by the semantic search service, it returns a list of URIs of individuals like the following:

```
<Q rdf:about="&portal;id1390">
  <degree>0.98</degree>
</Q>
<Q rdf:about="&portal;id0124">
  <degree>0.58</degree>
</Q>
```

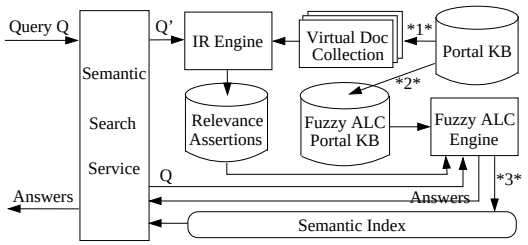

Figure 2: Semantic search service architecture

The value of the `degree` property is the *glb* value in our model. Currently, the implementation returns all results at once. Future implementation may use a more robust query answering dialogue and the OWL-QL[5] syntax.

Fig. 2 shows the internal architecture of the semantic search service. It shares the portal KB with the SPortS system. Just as the model is an integration of IR and formal query and reasoning, the implementation is an integration of an IR engine and a fuzzy-$\mathcal{ALC}$ reasoning engine. We use Lucene[6] as our IR engine and alc-F [28] as our fuzzy-$\mathcal{ALC}$ reasoning engine. The workflow of the semantic search service consists of two stages, the preprocessing stage and the answering stage.

In the preprocessing stage, three programs (shown as *1*, *2*, *3* respectively in Fig. 2) are run to build the virtual document collection, the fuzzy-$\mathcal{ALC}$ portal KB and the semantic index respectively. The first program uses the Jena API[7] to read the portal KB to construct the corresponding virtual document collection. For every node, the program finds all the nodes directly connected to it and then uses the simple concatenation method introduced in section 2.3 to build the text representation for it. We manually configure beforehand which property of each portal concept is used as the simple text representation for its individuals. For example, the system is configured to use the `title` property of the `Document` concept as simple text representations for all document nodes. The method of getting document node's full text content is also configured manually beforehand. In our case, every document node has a `URL` property from which the program can obtain the full text of the document. Non-textual files, such as .pdf, .doc and .ppt type files are further processed to extract their textual contents. After the text representations of every node are built, they are stored in the virtual document collection and the Lucene API is invoked to build a full text index for the collection.

From the original portal KB, a second program generates a fuzzy-$\mathcal{ALC}$ KB, which corresponds to the Σ_F in the formal description of the model. This is not a trivial task in the general case since the portal KB may be in a very expressive DL. In the current implementation, we are using the system for our lab's intranet portal which has a simple ontology and the second program simply ignores any concept constructor beyond $\mathcal{ALC}$ in the ontology. For the KB ABox, original assertions α are translated to $\langle \alpha \geq 1 \rangle$. In our scenario, this simple approach preserves most semantics of our lab's intranet portal ontology. It only loses some cardinality restrictions on roles.

[5]http://ksl.stanford.edu/projects/owl-ql/
[6]http://jakarta.apache.org/lucene/
[7]http://jena.sourceforge.net/

Finally, a third program builds an initial semantic index from the fuzzy-$\mathcal{ALC}$ KB. The semantic index is used later in the answering stage to boost the system's performance. We discuss it separately in section 4.1. All this preprocessing work can be done before the system starts accepting queries. Similar to the index information in the traditional IR, the preprocessing information needs to be updated periodically when the portal KB changes.

In the answering stage, the service starts accepting queries. For an accepted query Q, it first parses the query to obtain all the fuzzy IR concepts contained in it. IR queries (shown as Q' in Fig. 2) associated with these fuzzy IR concepts, if any, are issued to the Lucene engine to find relevant nodes from the virtual document collection. The result is a set of fuzzy assertions of the form $\langle i : Q' = x \rangle$ stating the degree of relevance from the IR perspective. They are stored as Relevance Assertions shown in Fig. 2. The union of the relevance assertions and the fuzzy-$\mathcal{ALC}$ KB corresponds to the Σ'_F KB in the formal description of the model. Σ'_F is then loaded into the fuzzy-$\mathcal{ALC}$ reasoning engine. Next, the semantic search service tries to calculate a $glb(\Sigma'_F, i : Q)$ value for every individual i in the KB and add to the result list those ones with glb values greater than 0. The result list is then sorted in descending order according to the glb values and returned.

A naive implementation of the semantic search service would rely on the fuzzy-$\mathcal{ALC}$ reasoning engine to calculate all the glb values. However, this is not necessary. Our implementation first consults the semantic index to obtain the glb values. It only delegates the task to the reasoning engine when necessary. This simple technique greatly reduces the number of reasoning tasks and can improve performance for most commonly asked conjunctive queries to the portal. We now give the details about the semantic index.

4.1 Semantic Index

Semantic index technique is used in optimizing DL instance retrieval problem [9]. Here, in our model, the intuition of the semantic index comes from the observation that the glb value of a conjunctive query can be derived from the glb values of its conjuncts. More precisely, given a fuzzy-$\mathcal{ALC}$ KB Σ, it can be proved that

$$glb(\Sigma, i : A \sqcap B) = min\{\, glb(\Sigma, i : A),\, glb(\Sigma, i : B)\,\}.$$

It follows that if we already know the glb values of A and B, we need not ask the reasoning engine for the new glb value of $A \sqcap B$. In addition, if A or B is an IR concept, the glb value is exactly the relevance value asserted in the IR Relevance Assertions.

Following this idea, we first build an initial semantic index in the preprocessing stage using the fuzzy-$\mathcal{ALC}$ reasoning engine. The index contains the glb values of all individuals w.r.t all named concepts in the fuzzy-$\mathcal{ALC}$ KB Σ_F. These glb values will remain the same in the Σ'_F KB[8]. In the implementation, the index is a hash table that maps a pair of individual and concept to a glb value. In the answering stage, if a conjunctive query is accepted, it is first decomposed to get its conjuncts. If a conjunct is an IR concept, the glb values are retrieved from the IR Relevance Assertions. If the glb values of the conjunct are already available in the semantic index, they are directly retrieved. Only those not available in the index are sent to the fuzzy-$\mathcal{ALC}$ reasoning engine for computation. The newly computed glb values can be cached in the semantic index for future retrieval. The caching strategy used in our implementation is to retain those mostly queried concepts in the index.

[8]Because fuzzy IR assertions are only related to new fuzzy IR concepts not appearing in Σ_F, they won't affect those glb values.

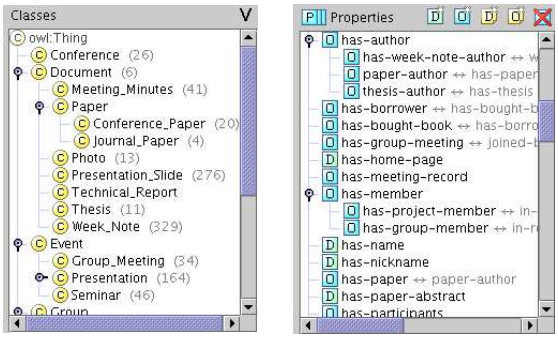

Figure 3: Lab intranet portal ontology

5. EVALUATION

As pointed out in [19], currently there is no commonly agreed evaluation methodology and benchmark for semantic search. In our work, we applied the semantic search service in our lab's intranet portal for searching and evaluation. The knowledge base and documents of our lab's intranet portal constitutes the evaluation dataset. The precision/recall measurement from the IR field is used as the basic evaluation criteria.

Our lab's intranet portal is powered by MS SharePoint portal server. By analyzing various information contained in the portal, we first manually built an ontology for the portal using Protégé[9]. Fig. 3 shows some concepts and roles of the ontology in Protégé. Individual information is then semi-automatically extracted from the portal's web pages to fill the ABox of the portal knowledge base. The final result is a portal knowledge base with about 35 concepts, 112 roles, 1400 individuals and 5000 ABox assertions. Among all the individuals, there are 687 documents with full text such as papers, presentation slides and meeting minutes etc. Based on the knowledge base, SPortS system can dynamically generate a semantic portal for ontology-based information browsing and editing. We intend to evaluate how our model performs in such a semantic portal.

As we have discussed in section 3.1, our model will degenerate to use pure IR method, pure formal query and reasoning method, and simple combination method to answer pure IR queries, pure formal queries, and simple conjunctive queries respectively. Therefore, our model will have the same performance as these methods for those kind of queries. What interests us, and also what we intend to evaluate, is how the model performs on queries that require deeper integration of IR and formal query and reasoning. For this purpose, we collected two information needs in our lab's intranet portal for evaluation. Imagine that a new graduate student joined our lab and would like to find all papers authored by some lab member related to ontology engineering. This is the information need 1 – IN1:

[9]http://protege.stanford.edu

	IN1	IN2
1	`Paper` ⊓ "ontology engineering"	`Paper` ⊓ "web search ontology engineering"
2	`Paper` ⊓ "ontology"	`Paper` ⊓ "search ontology"
3	`Paper` ⊓ "ontology engineering ORIENT"	`Paper` ⊓ "web search ontology ORIENT"

Table 1: Queries for simple combination method

IN1: find all papers authored by some lab member related to ontology engineering.

Later on, the student notices the "web search" research topic in our lab and is interested to know from papers what lab members related to ontology engineering might think about the topic. The student then refines the information need 1 to information need 2 – IN2:

IN2: find all papers talking about web search that are authored by some lab member related to ontology engineering.

Both of the two information needs require nontrivial combination of IR and formal query and reasoning. In the portal KB, `Conference_Paper` and `Journal_Paper` are subclasses of the `Paper` concept. Documents that are individuals of the two classes must be inferred to be papers. The authors of a paper can be obtained by querying the KB. Whether paper contents are related to web search and whether authors are related to ontology engineering should be handled by IR methods.

In order to evaluate our model's performance on satisfying the above two information needs, we adopted the classical evaluation criteria — precision and recall from the IR field. Precision/recall is an established standard evaluation criteria in the IR field that reports the system performance from the end user's point of view.

Because the pure IR method and the pure formal query and reasoning method can hardly support the above two information needs involving both IR and formal query and reasoning, we decide to compare our model's performance only with the simple combination method. In the simple combination method, queries are restricted to be a simple conjunction of a pure formal sub-query and a pure IR subquery. In this evaluation, we use our model to simulate the simple combination method by restricting the queries to be only of that form. Since it takes users' effort to formulate a complex query in our model, as a compensation we allowed the users to try and see the search results for several times to formulate the best query they deem appropriate for the simple combination method.

Table 1 shows the best queries used by three randomly-chosen subjects in the evaluation for the simple combination method. For the formal sub-query, they all use the `Paper` concept. For the IR sub-query, there are differences on which keywords to use among them. The third subject uses the keyword "ORIENT" because "ORIENT" is the name of a project developed by the ontology engineering group of our lab and the subject thinks the keyword brings more relevant results. All the subjects agree that these queries still can not fully express the information needs. However, in our model, the two information needs can be naturally expressed as the following two queries:

$QA1:$ $\texttt{Paper} \sqcap \exists \texttt{paper-author}.(\texttt{Lab\_Member} \sqcap D_q)$
$QA2:$ $(\texttt{Paper} \sqcap D_p) \sqcap \exists \texttt{paper-author}.(\texttt{Lab\_Member} \sqcap D_q)$

where D_p is a fuzzy concept for IR query p: *about web search* and D_q is a fuzzy concept for IR query q: *related to ontology engineering*. Naturally, we used the keywords "web search" and "ontology engineering" for D_p and D_q respectively. The two queries are sent to the semantic search service we developed to get the corresponding query results.

During the evaluation, a group of students of our lab are randomly chosen as the evaluation subjects. Every subject is first required to give his/her relevance judgements on the results of $QA1$ and $QA2$ w.r.t the two information needs IN1 and IN2. The subject is then allowed to use the simple combination method to do search for several times to find the best queries he/she deems appropriate. Because we use our model to simulate the simple combination method, the queries for simple combination method are actually also sent to the semantic search service we developed to get the results. The subject is then required to give the relevance judgements on the results returned for the best query he/she chooses for the simple combination method. The relevance judgements of the results of a query in the evaluation is later turned to a standard precision/recall curve. All such curves of all the subjects are then averaged to obtain the final precision/recall curves. Fig. 4 and Fig. 5 shows the final result. Table 2 compares the 11pt average precision.

The evaluation result shows that our model outperforms the simple combination method in both cases. The sim-

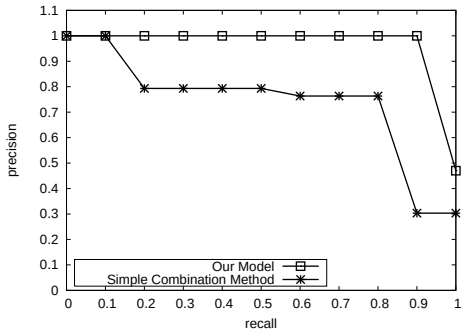

Figure 4: Precision/Recall for information need 1

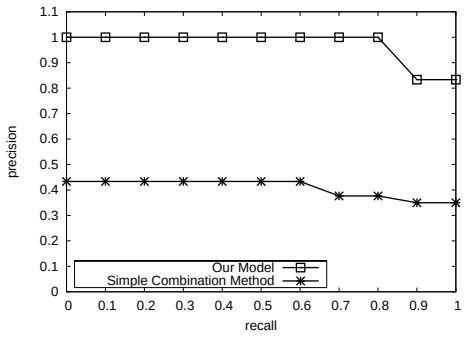

Figure 5: Precision/Recall for information need 2

	Simple Combination Method	Our Model
IN1	73.4%	95.2%
IN2	40.8%	96.9%

Table 2: 11pt average precision

ple combination method performs quite well for information need 1. However, for information need 2, which requires deeper integration of IR and formal query and reasoning, the simple combination method's performance drops very sharply while our model's performance consistently keeps high. Because of the lack of standard benchmark and large scale dataset, we are aware that the current evaluation is very limited. More thorough and larger scale evaluation is needed to further validate our claim. In the current evaluation, although we used the semantic index technique in the semantic search service, the query response time is still unacceptably long. This is due to the performance bottleneck of the fuzzy-$\mathcal{ALC}$ reasoning engine. As we have discussed in section 3.3, this model calls for a scalable fuzzy DL reasoning engine to solve the problem.

6. RELATED WORK

The ideas in the fuzzy DL based IR model [15] and the formal DL query method for the Semantic Web [11] are employed in our model. However, different from [15], we model an IR query as an atomic fuzzy IR concept while [15] uses SPS (Special Predicate Symbol) to abstract different functions of IR. For example, for the IR query "about future market", [15] represents it as $\exists$ ST.{future,market} where ST is the Similar Text SPS. We feel that the set after the ST predicate may not be appropriate for IR models that do not represent a query as a set of words. We thus use an atomic fuzzy IR concept to hide the details of the IR query. In addition, [15] is a pure IR model that searches only multimedia documents while our model extends to search also non-document individuals. [11] introduces conjunctive DL query method for querying the semantic web. However, it is a pure logical approach and uses only semantic information. Our model extends it to use and search also textual information.

Our work closely relates to the problem of question answering on the (Semantic) Web. [23] extends fuzzy DL for a web question answering system. Queries in it are simple concepts with property restrictions. The method uses heuristics to find texts within a web document that matches a query. It thus relies more on natural language processing techniques for query answering. Our model uses explicit semantics in a portal KB for query answering and the query can be any DL concept expression. As discussed in [14], there are actually a continuous spectrum of various methods for question answering on the Semantic Web. Our model can be seen as one that integrates IR and formal query and reasoning to utilize both textual and semantic information.

In the semantic search and retrieval area, [10, 21, 22, 18, 25, 1, 19] used semantic information to improve search on the (Semantic) Web. SHOE [10] collects semantic annotations on Web pages and stores them in a KB. Users are provided with a GUI to formulate an ontology-based query on the KB to find Web pages. When the query returns very few or no results, it provides a tool to automatically convert the formal query into a suitable IR query string for a Web search engine to find relevant pages. IR thus is used only as a complementary method for search. There is no tight integration of the two methods. Similar to SHOE, OWLIR [21] automatically generates and collects semantic annotations in web pages. It also performs reasoning to deduce more semantic information for a web page. The search request can contain both a formal query for semantic information and a keyword search for textual information. Web pages that satisfy both of them are returned as results. From the paper, it is hard to tell whether it provides other types of integration beyond the conjunction of IR query and formal query. The SCORE [22] system has a scalable semantic search engine that utilizes facts and metadata obtained from the Web via various text mining techniques. It lacks the formal reasoning capability as provided in our model, but it supports limited inferencing based on the traversal of relations in ontologies.

In TAP semantic search [18], search requests are primarily answered by a traditional web search engine. When the search request also matches some individual in the backend KB, structured semantic information about the individual from the KB is also presented to the end user. The two kinds of results are then displayed in one result page. This method uses only keywords as search request and does not provide formal query capability. It therefore also lacks a tight integration of the two methods. [1] presents the notion of semantic associations and describes methods for querying them on the semantic web. The notion of semantic associations captures the various ways in which two individuals in a knowledge base can be semantically related. However, it does not specify how to measure the relatedness of an individual w.r.t to a keyword search. Later on, [19] solves the problem by using the IR spreading activation method on a semantic network to find semantically relevant results for a given keyword search. The method is good at the discovery of indirect semantic relationships, but it lacks the formal query and reasoning capability. The spreading activation process is also hard to control for precision-oriented search. [25] discussed a ranking method for the Semantic Web that calculates the result relevance on the proof tree of a formal query. It is designed to be used with formal query method and does not incorporate IR relevance measurements.

Our work also relates to the integration of IR with the querying of (semi-)structured data. [7] proposed a probabilistic method for integrating fact and text retrieval. Facts and text terms are treated in the same way as attributes of an object. A query containing constrains on both facts and text contents can then be evaluated by calculating and combining the probabilities of the match of a (fact or text) attribute value with the query constraint on it. In a semantic portal, an object not only has attributes but also has relations to other objects and has types in a defined ontology. This information is hard to be utilized by this method. [8] combines probability theory with Datalog to model information retrieval. On the contrary, our model combines fuzzy theory with description logic and thus is based on a different formalism. Recently, there is also a large body of work on retrieving XML documents [30]. However, the use of IR in XML retrieval focuses more on combining IR with queries on XML tree structure. Because the lack of clear semantics of XML data, very few work exploits the semantic integration with IR. XSearch [4] relies on heuristics to find semantically relevant XML documents.

7. CONCLUSION AND FUTURE WORK

In this paper, we have presented an enhanced model that tightly integrates IR and formal query and reasoning for searching in semantic portals. The model extends the search capabilities of existing methods and can answer more complex search requests. Based on the model, a semantic search service is implemented and evaluated. The evaluation shows large improvements over exisiting methods.

As a future work, we plan to carry out a more large scale and thorough evaluation involving more queries. We are also going to explore other formalisms and methods for integrating IR and formal query and reasoning.

8. ACKNOWLEDGMENTS

The authors sincerely thank the anonymous reviewers for their valuable comments. We also would like to thank Min Long, Shenghua Bao and Xian Wu for helping build the semantic search service and carry out the evaluation.

9. REFERENCES

[1] K. Anyanwu and A. Sheth. The r-operator: Enabling querying for semantic associations on the semantic web. In *Proc. of WWW 2003*, Budapest, Hungary, 2003.

[2] N. Athanasis, V. Christophides, and D. Kotzinos. Generating on the fly queries for the semantic web: The ICS-FORTH graphical RQL interface (GRQL). In *Proc. of ISWC 2004*.

[3] T. Catarci, T. D. Mascio, E. Franconi, G. Santucci, and S. Tessaris. An ontology based visual tool for query formulation support. In *Proc. of ECAI 2004*.

[4] S. Cohen, J. Mamou, Y. Kanza, and Y. Sagiv. XSearch: A semantic search engine for XML. In *Proc. of 29th Intl. Conf. on Very Large Data Bases (VLDB2003)*, pages 45–56, Berlin, Germany, 2003.

[5] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[6] D.E.Losada and A.Barreiro. Propositional logic representations for documents and queries: A large-scale evaluation. In *Proc. of ECIR2003*, Pisa, Italy, 2003.

[7] N. Fuhr. Integration of probabilistic fact and text retrieval. In *Proc. of the ACM SIGIR'92*, pages 211–222, 1992.

[8] N. Fuhr. Probabilistic Datalog: Implementing logical information retrieval for advanced applications. *Journal of the American Society for Information Science and Technology*, 51(2):95–110, 2000.

[9] V. Haarslev and R. Möller. Optimization strategies for instance retrieval. In *Intl. Workshop on Description Logics*, 2002.

[10] J. Heflin and J. Hendler. Searching the Web with SHOE. In *Proc. of AAAI-2000 Workshop on AI for Web Search*, 2000.

[11] I. Horrocks and S. Tessaris. Querying the semantic web: A formal approach. In *Proc. of ISWC 2002*, pages 177–191, 2002.

[12] H. Lausen, M. Stollberg, R. L. Hernández, Y. Ding, S.-K. Han, and D. Fensel. Semantic Web Portals - state of the art survey. Technical Report TR-2004-04-03, DERI(www.deri.org), 2004.

[13] C. Lin, L. Zhang, J. Zhou, Y. Yang, and Y. Yu. SPortS: Semantic+Portal+Service. In *ECAI 2004 Workshop on Application of Semantic Web Technologies to Web Communities*, volume 107 of *CEUR-WS*, 2004.

[14] D. L. McGuinness. Question answering on the semantic web. *IEEE Intelligent Systems*, 19(1), 2004.

[15] C. Meghini, F. Sebastiani, and U. Straccia. A model of multimedia information retrieval. *Journal of the ACM*, 48(5):909–970, 2001.

[16] C. Meghini, F. Sebastiani, U. Straccia, and C. Thanos. A model of information retrieval based on a terminological logic. In *Proc. of ACM SIGIR'93*, Pittsburgh, USA, 1993.

[17] C. Meghini and U. Straccia. A relevance terminological logic for information retrieval. In *Proc. ACM SIGIR'96*, pages 197–205, Zurich,Switzerland, 1996.

[18] R.Guha, R. McCool, and E. Miller. Semantic search. In *Proc. of WWW 2003*, 2003.

[19] C. Rocha, D. Schwabe, and M. P. de Aragão. A hybrid approach for searching in the semantic web. In *Proc. of WWW 2004*, pages 374–383, 2004.

[20] F. Sebastiani. A probabilistic terminological logic for modelling information retrieval. In *Proc. of ACM SIGIR'94*, Dublin, Ireland, 1994.

[21] U. Shah, T. Finin, A. Joshi, R. Cost, and J. Mayfield. Information retrieval on the semantic web. In *Proc. of the 11th Intl. Conf. on Information and Knowledge Management (CIKM2002)*, pages 461–468, 2002.

[22] A. Sheth, C. Bertram, D. Avant, B. Hammond, K. Kochut, and Y. Warke. Managing semantic content for the Web. *IEEE Internet Computing*, 6(4), 2002.

[23] S. Singh, L. Dey, and M. Abulaish. A framework for extending fuzzy description logic to ontology based document processing. In *Proc. 2nd Intl Atlantic Web Intelligence Conf (AWIC 2004)*, 2004.

[24] S. Staab, J. Angele, S. Decker, M. Erdmann, A. Hotho, A. Maedche, R. Studer, and Y. Sure. Semantic community web portals. In *Proc. of WWW9*.

[25] N. Stojanovic, R. Studer, and L. Stojanovic. An approach for the ranking of query results in the semantic web. In *Proc. of ISWC 2003*, 2003.

[26] U. Straccia. Reasoning within fuzzy description logics. *Journal of Artificial Intelligence Research*, 14, 2001.

[27] U. Straccia. Transforming fuzzy description logics into classical description logics. In *Proc. of the 9th European Conference on Logics in Artificial Intelligence (JELIA-04)*, 2004.

[28] U. Straccia and A. Lopreiato. alc-F: A fuzzy ALC reasoning engine, 2004. http://faure.iei.pi.cnr.it/~straccia/software/alc-F/.

[29] C. Tempich and R. Volz. Towards a benchmark for Semantic Web reasoners - an analysis of the DAML ontology library. In *ISWC2003 Workshop on Evaluation of Ontology-Based Tools*, 2003.

[30] R. W.P.Luk, H.V.Leong, T. S.Dillon, A. T.S.Chan, W. Croft, and J. Allan. A survey in indexing and searching XML documents. *JASIST*, 53(6):415–437, 2002.