Semantic Matching of Natural Language Web Queries

Naouel Karam[1], Salima Benbernou[2], Mohand-Said Hacid[2], and Michel Schneider[1]

[1] LIMOS (Laboratoire d'Informatique, de Modélisation et d'Optimisation des Systèmes)
Université Blaise Pascal Clermont2,
Complexe Scientifique des Cézeaux 63177 Aubière, France
{karam, schneider}@isima.fr
[2] UFR Informatique
Université Claude Bernard Lyon1,
43 bd du 11 novembre 69100 Villeurbanne, France
{salima.benbernou, mshacid}@bat710.univ-lyon1.fr

Abstract. In this paper, we propose a method to automatically rank documents returned by a search engine in the WWW with respect to a query. The process consists in three steps, the first translates the query and document descriptions into description logic terminologies. The second computes a mapping between related elements in the query and each document. This mapping matches concepts in the terminologies based on their names and their definitions. The last step computes the difference between the query (represented as a terminology) and each document (also represented as a terminology) and ranks the documents according this difference. To deal with linguistic information when comparing description logic concepts, we propose a definition of subsumption that takes into account names similarity between concepts occurring in the descriptions being compared. We describe each step of the method and show the intended results on a running example.

1 Introduction

This paper deals with the problem of ranking documents returned by a search engine in the WWW. The ranking function involves a semantic comparison between the client query and the documents content. The document that best matches the query is returned first. To perform this ranking we need:

- A formal description of the query and the documents content: the query and the documents are specified in natural language (NL), a formal description of their semantics is needed to achieve an automatic comparison.
- A matching algorithm that compares a query description and a document description and returns a set of matching elements between the query and the document.
- A ranking function that sorts the documents with respect to the size of the part of the query that is not covered by the documents.

N. Koch, P. Fraternali, and M. Wirsing (Eds.): ICWE 2004, LNCS 3140, pp. 416–429, 2004.
© Springer-Verlag Berlin Heidelberg 2004

The Proposed Approach

We propose to use description logics (DLs) [2] as a formal representation language for specifying documents and queries. The motivations are twofold: DLs come with well-defined semantics and correct inference algorithms and the formalization of a text in DLs has already been studied, see [8] for details about how we extract a terminology from a natural language document. As already stated in [14], not all aspects of a natural language can be captured by a formal description, we will restrict ourselves to a small fragment of NL that can be represented in DLs.

The matching step consists in comparing the two terminologies obtained from a query and a document. Given two terminologies $\mathcal{T}_Q$ and $\mathcal{T}_D$ describing a query Q and a document D respectively, our goal is to find the elements in $\mathcal{T}_Q$ and $\mathcal{T}_D$ that match. This is done by a *matching* function that takes two terminologies as input and produces a one to one *mapping* between defined concepts of the two terminologies that correspond semantically to each other.

Finally, the documents are ranked according to the size of the extra information contained in the query and not in the documents. The extra information is calculated with the help of the difference operation between pairs of mapped elements. We propose an algorithm that computes the difference between $\mathcal{ALE}$-concept descriptions. It is based on the work reported in [10], by taking into account linguistic relations (synonymy, hypernymy...) between concept names occurring in the two descriptions.

The paper is organized as follows. Section 2 presents the motivation behind this work. Section 3 gives a brief overview of description logics and the difference operator. We define the matching and the ranking problems in sections 4 and 5 respectively. We conclude in section 6.

2 Motivation

All major search engines available on the web rank web pages by determining relevancy through analyzing keyword location, frequency and through other methods, for example, by analyzing how pages link to each other. Non relevant pages appear frequently in the resulted rank and it may take time for the user finding out the intended information among this huge number of pages.

Our goal is to allow the user to describe his requirements by specifying a detailed description so that we can compare it semantically to the content of web pages. The most relevant page compared to the query needs comes first.

The proposed algorithm detects the parts of the documents that are semantically related to some parts of the query and then deduces the non covered part in the query. The most relevant match will be the one with the smallest non covered part.

Let us illustrate the practical interest of our method with an example.

Example 1. Consider the three simple NL texts depicted in Figure 1. The query Q describes the rooms of a hotel, D_1 and D_2 are documents returned by a research engine.

Q	All the rooms are comfortable and air conditioned. Each room is provided with a TV and a large bed. The hotel is located in Paris.
D_1	Located in the french capital, the hotel is convivial. Each bedroom is air conditioned and provided with a cable television. The breakfast is served in an elegant lounge.
D_2	The hotel is located in Paris. All our homelike rooms are provided with a queen size bed, a color TV and a private bathroom. The hotel accepts only credit card guarantee.

Fig. 1. Simple NL texts

By reading the documents, a user can see that the second document matches better the query needs than the first one. To do this, we need to be able to detect the related parts between the query and each document. The first document shares with the query the same information about hotel location, air conditioning and the existence of a television. For the second document, the common information concerns hotel location, comfort of the rooms and the existence of a television and a large bed.

This discovery is challenging because different words can be used to express the same semantic information. This can be done by synonyms (e.g. *comfortable* vs *homelike*, *french capital* vs *Paris*) or hypernyms (e.g. *bedroom* vs *room*, *cable television* vs *TV*).

Once this matching performed, the extra information contained in the query and not in the document can be computed easily. It is clear that the document that better meets the user needs is the one with the minimal extra information.

3 Preliminaries

In this section, we introduce description logics, the formalism used in our framework, the difference operator and the notion of size of a description.

3.1 Description Logics

Description logics (DLs, also called terminological logics) are a family of knowledge representation formalisms designed for representing and reasoning about terminological knowledge. In DLs, the conceptual knowledge of an application domain is represented in terms of *concepts* (unary predicates) that are interpreted as sets of individuals, and *roles* (binary predicates) that are interpreted as binary relations between individuals.

Starting with the set N_C of concept names and the set N_R of role names, complex concept descriptions are built inductively using concept constructors. The different description logic languages distinguish themselves by the kind of constructs they allow. In our framework we are going to use the $\mathcal{ALE}$ description logic. In this description logic, concept descriptions are formed according to the syntax rules depicted Figure 2. $A \in N_C$ denotes a concept name, $r \in N_R$ a role name, and C, D (complex) concept descriptions.

$C, D \rightarrow \top \mid$	(top-concept)
$\bot \mid$	(bottom-concept)
$A \mid$	(concept name)
$\neg A \mid$	(primitive negation)
$C \sqcap D \mid$	(conjunction)
$\exists r.C \mid$	(existential restriction)
$\forall r.C \mid$	(value restriction)

Fig. 2. Syntax of some concept descriptions

Let $\mathcal{L}$ denotes some description logic, a concept built using the constructors of $\mathcal{L}$ is called an $\mathcal{L}$-concept.

The semantics of a concept description is defined by the notion of interpretation as given below.

Definition 1. *(Interpretation) An interpretation $\mathcal{I} = (\triangle^{\mathcal{I}}, .^{\mathcal{I}})$ consists of a non-empty set $\triangle^{\mathcal{I}}$, the domain of the interpretation, and an interpretation function $.^{\mathcal{I}}$ that maps each concept name $A \in N_C$ to a subset of $\triangle^{\mathcal{I}}$ and each role name $r \in N_R$ to a binary relation $r^{\mathcal{I}}$, subset of $\triangle^{\mathcal{I}} \times \triangle^{\mathcal{I}}$. The interpretation function can be extended to arbitrary concept descriptions as shown in Figure 3.*

$$
\begin{aligned}
\top^{\mathcal{I}} &= \Delta^{\mathcal{I}} \\
\bot^{\mathcal{I}} &= \emptyset \\
(\neg A)^{\mathcal{I}} &= \Delta^{\mathcal{I}} \setminus A^{\mathcal{I}} \\
(C \sqcap D)^{\mathcal{I}} &= C^{\mathcal{I}} \cap D^{\mathcal{I}} \\
(\exists r.C)^{\mathcal{I}} &= \{x \in \Delta^{\mathcal{I}} \mid \exists y : (x, y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \\
(\forall r.C)^{\mathcal{I}} &= \{x \in \Delta^{\mathcal{I}} \mid \forall y : (x, y) \in r^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}
\end{aligned}
$$

Fig. 3. Semantics of concept descriptions

DL systems provide various reasoning services, the most important is the computation of the subsumption relation.

Definition 2. *(Subsumption) Let C, D be concept names, D subsumes C (noted $C \sqsubseteq D$) iff $C^I \subseteq D^I$ for all interpretation $\mathcal{I}$.*

Concept descriptions are used to specify terminologies that define the intentional knowledge of an application domain.

Definition 3. *(Terminology) Let A be a concept name and C a concept definition. Then $A \doteq C$ and $A \sqsubseteq C$ are terminological axioms. The first is a complete definition, the second an incomplete one. A terminology $\mathcal{T}$ is a finite set of terminological axioms such that no concept name appears more than once in the left-hand side of a definition. If a concept A occurs in the left-hand side of a definition, it is called* defined concept*. The other concepts are called* primitive concepts.

A terminology built using the constructors of some description logic $\mathcal{L}$ is called an $\mathcal{L}$-terminology.

An interpretation $\mathcal{I}$ is a model of a terminology $\mathcal{T}$ if it satisfies all the statements contained in $\mathcal{T}$:
- $A^{\mathcal{I}} = C^{\mathcal{I}}$ for all terminological axioms $A \doteq C$ in $\mathcal{T}$,
- $A^{\mathcal{I}} \subseteq C^{\mathcal{I}}$ for all terminological axioms $A \sqsubseteq C$ in $\mathcal{T}$.

In our work, natural language documents are represented by a DL terminology, NL statements are transformed into terminological axioms.

Example 2. The NL texts of example 1 are represented by the $\mathcal{ALE}$-terminologies $\mathcal{T}_Q$, $\mathcal{T}_{D_1}$ and $\mathcal{T}_{D_2}$ given in Figure 4.

$\mathcal{T}_Q$	Room	$\doteq$ Comfortable $\sqcap$ Air-conditioned $\sqcap$ $\exists$provided-with.TV $\sqcap$ $\exists$provided-with.(Bed $\sqcap$ $\exists$has-size.Large) $\sqcap$ $\overline{\text{Room}}_Q$
	Hotel	$\doteq$ $\exists$located-in.Paris $\sqcap$ $\overline{\text{Hotel}}_Q$
$\mathcal{T}_{D_1}$	Hotel	$\doteq$ $\exists$located-in.French-capital $\sqcap$ Convivial $\sqcap$ $\overline{\text{Hotel}}_{D_1}$
	Bedroom	$\doteq$ Air-conditioned $\sqcap$ $\exists$provided-with.Cable-television $\sqcap$ $\overline{\text{Bedroom}}_{D_1}$
	Breakfast	$\doteq$ $\exists$served-in.(Elegant $\sqcap$ Lounge) $\sqcap$ $\overline{\text{Breakfast}}_{D_1}$
$\mathcal{T}_{D_2}$	Hotel	$\doteq$ $\exists$located-in.Paris $\sqcap$ $\forall$accept.Credit-card-guarantee $\sqcap$ $\overline{\text{Hotel}}_{D_2}$
	Room	$\doteq$ Homelike $\sqcap$ $\exists$provided-with.(Bed $\sqcap$ $\exists$has-size.Queen-size) $\sqcap$ $\exists$provided-with.Color-TV$\sqcap\exists$provided-with.(Bathroom$\sqcap$Private)$\sqcap$ $\overline{\text{Room}}_{D_2}$

Fig. 4. Examples of terminologies

The overlined concepts stand for the missing part of the definitions, we will ignore these concepts in the rest of the paper.

3.2 The Difference Operator

Informally speaking, the difference between two concept descriptions is the information contained in the first description and not in the second. The difference operator allows to remove from a given description all the information contained in another description. The difference operation between two concept descriptions was first introduced by Teege [15]. The difference between two concept descriptions C and D with $C \sqsubseteq D$ is given by

$$C - D := max\{E \mid E \sqcap D \equiv C\}$$

where max is defined with respect to subsumption.

[10] proposed a refinement of this definition by allowing the difference between incomparable descriptions (i.e. D is not required to subsume C) and taking the syntactic minimum (w.r.t a subdescription ordering $\preceq_d$) instead of a semantic maximum. The difference between two incomparable concept descriptions C and D is defined as

$$C - D := min\{E \mid E \sqcap D \equiv C \sqcap D\}$$

where *min* is defined with respect to a subdescription ordering.

This definition has two advantages, it does not contain redundancies and it is more readable by a human user. However, Tegee's difference captures the real semantic difference between two concept descriptions.

We use the second definition because it is defined for the DL $\mathcal{ALE}$ that allows us to represent a considerable number of NL semantics.

3.3 Size of a Concept Description

We define the size $|C|$ of an $\mathcal{ALE}$-concept description C as the number of conjuncts occurring on the top-level of C.

Example 3. The sizes of the concepts *Room* and $Hotel$ of $\mathcal{T}_Q$ in Example 2 are 5 and 2 respectively.

4 The Matching Algorithm

In this section we introduce the matching operation. *Match* is an operation that takes a query description and a document description and returns a *mapping* that identifies corresponding elements in the two descriptions. This *mapping* consists of a set of mapping elements indicating that certain elements of the query Q are related to certain elements of the document D. By elements, we mean the defined concepts in the terminologies $\mathcal{T}_Q$ and $\mathcal{T}_D$. A concept A_i from $\mathcal{T}_Q$ is related to a concept B_i from $\mathcal{T}_D$ if their names and their definitions are similar.

In [11], a schema matching algorithm called Cupid was proposed. A schema consists of a set of related elements such as database or XML elements. The result of the match operation is a mapping indicating that certain elements of the first schema are related to certain elements of the second. Similarity coefficients are computed between elements of two schemas in two phases, a linguistic and a structural one. Then a mapping is deduced from those coefficients.

Following Cupid intuitions, we build an algorithm for matching a query description and a document description. First, it proceeds by computing similarity coefficients between defined concepts in the two terminologies and then deduces a mapping from those coefficients. The coefficients in the range [0,1] are calculated in two steps:

- **Step 1.** Matching of names : it is based on the notion of semantic relatedness introduced in [7] that measures the extent to which two lexicalized concepts are close. This measure is based on the semantic relations of Wordnet [6]. We will call the result the *name similarity coefficient* ($nsim$).
- **Step 2.** Matching of description. it consists in comparing the concept descriptions occurring in the two terminologies. This phase uses name similarities between concepts appearing in the concept descriptions. We will call the result the *description similarity coefficient* ($dsim$).

The weighted similarity ($wsim$) is a mean of $nsim$ and $dsim$, it is calculated as follows: $wsim = w \times nsim + (1 - w) \times dsim$, where w is a constant in the range [0,1]. We compute weighted similarity coefficients between defined concepts in the terminologies. A mapping ρ is deduced from those coefficients by choosing pairs of elements with maximal weighted similarity.

In the next two subsections, we detail name matching and description matching steps.

4.1 Name Matching

The first step of the matching is based on defined concept names. We need to determine the degree of semantic similarity between two concept names. We reuse the notion of semantic relatedness between two lexically expressed concepts [7]. This measure uses WordNet [6] as knowledge source. The idea behind this measure is that two concepts are close if the path relating them in WordNet is not long and does not change direction too often. We recall the definition of semantic relatedness and define the name similarity coefficient. It is expressed as function of the semantic relatedness since we require it to be in the range [0,1].

Definition 4. *(Semantic relatedness)[7] The semantic relatedness of two concept names c_1 and c_2 is given by:*

$$rel(c_1, c_2) = C - PathLength(c_1, c_2) - k * NumberOfChangesOfDirection(c_1, c_2),$$

where C and k are constants and PathLength denotes the length of the shortest path between two concepts. If no such path exists, $rel(c_1, c_2)$ is zero.

Definition 5. *(Name similarity coefficient) The name similarity of two concept names P_1, $P_2 \in N_C$ is given by:*

$$nsim(P_1, P_2) = \frac{rel(P_1, P_2)}{C}.$$

where C is the same constant used in the definition of the semantic relatedness.

4.2 Description Matching

The intuition behind the description matching is that two concept descriptions are similar if their difference is minimal. We estimate the description similarity coefficient as a function of the size of the difference between the two descriptions.

Definition 6. *(Description similarity coefficient) The description similarity between two concept descriptions C and D is given by:*

$$dsim(C, D) = 1 - \frac{|C - D|}{|C|}$$

The algorithm proposed in [10] that performs the difference between two $\mathcal{ALE}$-concept descriptions is based on the subsumption test. We propose a new definition of the subsumption, that takes into account linguistic information about concepts occurring in the descriptions being compared. It is denoted by $\sqsubseteq_{\mathcal{S}}$. In order to exploit the graph-based subsumption reasoning, we are going to represent the concepts occurring in the terminologies as trees.

We know that a tree-based characterization of subsumption was stated in [1]. It works in three steps. First, concept descriptions are turned into normal forms, that makes the knowledge implicitly contained in a concept description explicit. Second, these normal forms are translated into description trees. Then subsumption is characterized in terms of graph homomorphism between the description trees.

We first recall the definition of normal forms and description trees, then we propose a definition of homomorphism taking into account name equivalence between concept names occurring in the descriptions.

Definition 7. *($\mathcal{ALE}$-normalization rules) Let C, D be two $\mathcal{ALE}$-concept descriptions and $r \in N_R$ a primitive role. The $\mathcal{ALE}$-normalization rules are defined as follows*

$$\begin{aligned}
\forall r.C \sqcap \forall r.D &\rightarrow \forall r.(C \sqcap D)\\
\forall r.C \sqcap \exists r.D &\rightarrow \forall r.C \sqcap \exists r.(C \sqcap D)\\
\forall r.\top &\rightarrow \top\\
C \sqcap \top &\rightarrow C\\
P \sqcap \neg P &\rightarrow \bot, \textit{for each } P \in N_C\\
\exists r.\bot &\rightarrow \bot\\
C \sqcap \bot &\rightarrow \bot
\end{aligned}$$

If only the rule $\forall r.\top \rightarrow \top$ is applied to a concept description C, the resulting concept is called in $\top$-normal form and the corresponding description tree is noted $\mathcal{G}_C^{\top}$.

Definition 8. *($\mathcal{ALE}$-description trees) An $\mathcal{ALE}$-description tree is a tree of the form $\mathcal{G} = (N, E, n_0, \ell)$ where*

- *N is a finite set of nodes of $\mathcal{G}$;*
- *$E \subseteq N \times (N_R \cup \forall N_R) \times N$ is a finite set of edges labeled with role names r ($\exists$-edges) or with $\forall r$ ($\forall$-edges); $\forall N_R := \{\forall r \mid r \in N_R\}$;*
- *n_0 is the root of $\mathcal{G}$;*
- *ℓ is a labeling function mapping the nodes in N to finite sets $\{P_1, ..., P_k\}$ where each P_i, $1 \leq i \leq k$, is one of the following forms: $P_i \in N_C$, $P_i = \neg P$ for some $P \in N_C$, or $P_i = \bot$. The empty label corresponds to the top-concept.*

For $n, m \in N$ and $r \in N_R$, an $\exists$-edge from n to m labeled r is written as nrm, and a $\forall$-edge as $n\forall rm$.

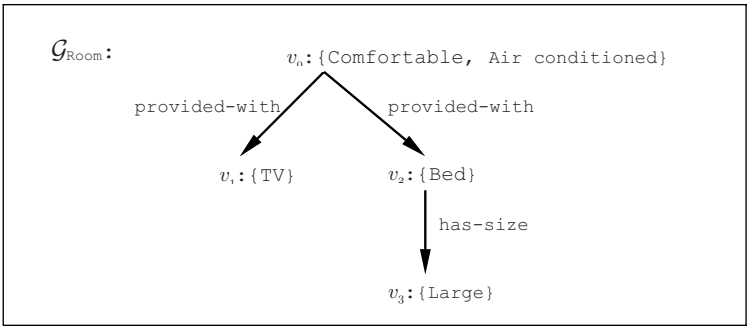

Fig. 5. an $\mathcal{ALE}$-description tree

Every $\mathcal{ALE}$-concept description C can be turned into an $\mathcal{ALE}$-description tree $\mathcal{G}_C$ (see [1] for a formal definition of this translation).

Example 4. The $\mathcal{ALE}$-concept description

$$\text{Room} \doteq \text{Comfortable} \sqcap \text{Air-conditioned} \sqcap \exists\text{provided-with.TV} \sqcap \exists\text{provided-with.(Bed} \sqcap \exists\text{has-size.Large)}$$

yields the tree $\mathcal{G}_{Room}$ of Figure 5.

Definition 9. *(Name equivalence) Let P_1 and P_2 be two concept names in N_C, P_1 and P_2 are said to be equivalent, written $P_1 \equiv P_2$, if $nsim(P_1, P_2)$ exceeds a certain threshold th_{sim}.*

Given a set $\mathcal{S}$ of name equivalences between concept names and two $\mathcal{ALE}$-description trees, we define the notion of homomorphism between description trees w.r.t a name equivalence set as follows.

Definition 10. *(Homomorphism on description trees w.r.t a name equivalence set) A mapping $\varphi : N_H \rightarrow N_G$ from an $\mathcal{ALE}$-description tree $\mathcal{H} = (N_H, E_H, m_0, \ell_H)$ to an $\mathcal{ALE}$-description tree $\mathcal{G} = (N_G, E_G, n_0, \ell_G)$ is called homomorphism w.r.t a name equivalence set $\mathcal{S}$, if and only if the following conditions are satisfied:*

1. *$\varphi(m_0) = n_0$;*
2. *for all $n \in N_H$ we have, $\forall P_i \in \ell_H(n), \exists P_j \in \ell_G(\varphi(n))$ such that $P_i \equiv P_j$ is in $\mathcal{S}$ or $\bot \in \ell_G(\varphi(n))$;*
3. *for all $nrm \in E_H$, either $\varphi(n)r\varphi(m) \in E_G$, or $\varphi(n) = \varphi(m)$ and $\bot \in \ell_G(\varphi(n))$; and*
4. *for all $n\forall rm \in E_H$, either $\varphi(n)\forall r\varphi(m) \in E_G$, or $\varphi(n) = \varphi(m)$ and $\bot \in \ell_G(\varphi(n))$.*

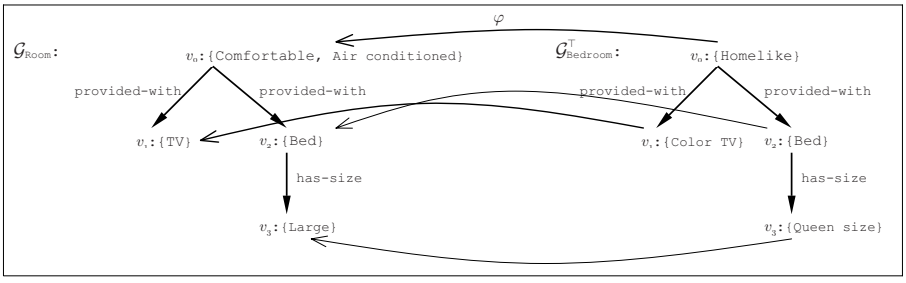

Fig. 6. Subsumption between $\mathcal{ALE}$-description trees

Theorem 1. *Let C, D be $\mathcal{ALE}$-concept descriptions. Then, $C \sqsubseteq_{\mathcal{S}} D$ iff there exists a homomorphism w.r.t a name equivalence set $\mathcal{S}$ from $\mathcal{G}_D^{\top}$ to $\mathcal{G}_C$.*

Sketch of proof. The proof is based on the one given for the theorem 41 in [1]. The idea is to use at different stages of the proof the fact that if $P \equiv P'$ we have $x_0 \in P^I$ implies that $x_0 \in P'^I$ and vice versa.

Example 5. Let us illustrate Theorem 1 by two concept descriptions, namely Room and Bedroom

$$\begin{aligned}\text{Room} \doteq\ & \text{Comfortable} \sqcap \text{Air-conditioned} \sqcap \exists\text{provided-with.TV} \sqcap \\ & \exists\text{provided-with.(Bed} \sqcap \exists\text{has-size.Large)} \\ \text{Bedroom} \doteq\ & \text{Homelike} \sqcap \exists\text{provided-with.Color TV} \sqcap \exists\text{provided-with.(Bed} \sqcap \\ & \exists\text{has-size.Queen size)}\end{aligned}$$

and the name equivalence set $\mathcal{S}$

$$\begin{aligned}\mathcal{S} = \{&\text{Comfortable} \equiv \text{Homelike}, \\ &\text{TV} \equiv \text{Color TV}, \\ &\text{Large} \equiv \text{Queen size}\}\end{aligned}$$

The descriptions are already in a normal form. A homomorphism from $\mathcal{G}_{Bedroom}$ to $\mathcal{G}_{Room}$ w.r.t $\mathcal{S}$ is depicted in Figure 6. From Theorem 1 we can conclude that $Room \sqsubseteq_{\mathcal{S}} Bedroom$.

vvvvvvvv

Based on the algorithm proposed in [10] for computing the difference between $\mathcal{ALE}$-concept descriptions, we propose the algorithm diff_{sim} depicted in Figure 7. Two changes have been made to the original algorithm. First, the definition of prim(C-D) has been changed since we are dealing with a name equivalence operator instead of equality of concept names. Second, the proposed subsumption over a set of name equivalence is used instead of the classical subsumption (line 6). The following notations are used:

- $prim(C)$ denotes the set of (negated) concept names and the bottom concept occurring on the top-level of C,
- $C.r = E$ if there exists a value restriction $\forall r.E$ on the top-level of C; $C.r = \top$ otherwise,
- $\exists r.C' \in C$ means that $\exists r.C'$ occurs on the top-level of C.

Require: $\mathcal{ALE}$-concept descriptions C and D in $\mathcal{ALE}$-normal form, a set $\mathcal{S}$ of name equivalences.
Ensure: $\text{diff}_{sim}(C, D)$
if $C \sqcap D \equiv \bot$ **then**
  $\text{diff}_{sim}(C, D) := \bot$
else
  $\text{diff}_{sim}(C, D) := \sqcap_{A \in prim(C-D)} A \sqcap \forall r.\text{diff}_{sim}(C.r, D.r) \sqcap \sqcap_{E \in \mathcal{E}'_r} \exists r.E$
  where $prim(C - D) := \{P \in prim(C) \mid$ there does not exist $P' \in prim(D)$ with $P \equiv P' \in \mathcal{S}\}$ and the value restriction is omitted in case $\text{diff}_{sim}(C.r, D.r) \equiv \bot$ and $\mathcal{E}'_r$ is computed as follows:
  Let $\exists r.C_1, ..., \exists r.C_n \in C, \exists r.D_1, ..., \exists r.D_m \in D$ be all the existential restrictions in the top level of C and D, respectively, $\mathcal{E}_r = \{C_1, ..., C_n\}$.
  for $i = 1$ to n **do**
    if (i) there exists $j \in \{1, ..., n\}, j \neq i$,with $D.r \sqcap C.r \sqcap C_j \sqsubseteq_{\mathcal{S}} C_i$, or
    (ii) there exists $j \in \{1, ..., m\}$,with $D.r \sqcap C.r \sqcap D_j \sqsubseteq_{\mathcal{S}} C_i$, **then**
      $\mathcal{E}_r := \mathcal{E}_r \backslash \{C_i\}$
    end if
  end for
  $\mathcal{E}'_r = \{E^* \mid E \in \mathcal{E}_r\}$ where $E^* := \text{diff}_{sim}(E, C.r \sqcap D.r)$.
end if

Fig. 7. The algorithm diff_{sim}

Example 6. Computing the difference between the concept descriptions of Example 5 yields

$$\text{Room} - \text{Bedroom} = \text{Air-conditioned}$$

4.3 Mapping Generation

The set of mapping elements is deduced from the computed name and description similarities. Let $\mathcal{T}_Q = \{Q_i \doteq C_i, i \in [1, n]\}$ and $\mathcal{T}_D = \{D_j \doteq C_j, j \in [1, m]\}$ be two $\mathcal{ALE}$-terminologies describing a query Q and a document D respectively. A mapping ρ from $\mathcal{T}_Q$ to $\mathcal{T}_D$ is computed as follows:

- $\rho(Q_i) = D_j$, $1 \leq i \leq n$, $1 \leq j \leq m$, if $wsim(Q_i, D_j) \geq th_{map}$ and $wsim(Q_i, D_j) > wsim(Q_i, D_k)$ for all $D_k \in \mathcal{T}_D$, $k \neq j$,
- $\rho(Q_i) = \top$, if there is no $D_j, 1 \leq j \leq m$, with $wsim(Q_i, D_j) \geq th_{map}$.

Example 7. Let us illustrate the matching step on the terminologies $\mathcal{T}_Q$ and $\mathcal{T}_{D_1}$ of Example 2. The set of computed mappings is depicted in Table 1. The name similarity coefficients $nsim$ are computed with $C = 8$ and $k = 1$. The chosen thresholds th_{sim} and th_{map} are 0.75 and 0.5 respectively. For $wsim$, we take $w = 0.5$.

Table 1. The mapping generated from $\mathcal{T}_Q$ and $\mathcal{T}_{D_1}$

ρ	$nsim$	$dsim$	$wsim$
Room → Bedroom	0.87	0.5	0.68
Hotel → Hotel	1	1	1

5 The Ranking Problem

In this section we show how the mapping generated by the matching algorithm is used to compute the difference between a query terminology and a document terminology. Then we show how documents are ranked with respect to this difference.

Let $\mathcal{T}_Q = \{Q_i \doteq C_i, i \in [1,n]\}$ and $\mathcal{T}_D = \{D_j \doteq C_j, j \in [1,m]\}$ be two $\mathcal{ALE}$-terminologies describing a query Q and a document D respectively. The difference between the two terminologies is defined as follows:

Definition 11. *(Difference between terminologies) Given the mapping ρ resulting from the matching between $\mathcal{T}_Q$ and $\mathcal{T}_D$. The difference between the terminologies $\mathcal{T}_Q$ and $\mathcal{T}_D$ is the conjunction of the differences between each related pair of concepts in the two terminologies.*

$$diff_\rho(\mathcal{T}_Q, \mathcal{T}_D) = \sqcap_{Q_i \in \mathcal{T}_Q}(Q_i - \rho(Q_i))$$

With the notion of size of a description, we define the dissimilarity coefficient between two terminologies.

Definition 12. *(Dissimilarity coefficient) The dissimilarity coefficient between the terminologies $\mathcal{T}_Q$ and $\mathcal{T}_D$ is the size of their difference*

$$d(\mathcal{T}_Q, \mathcal{T}_D) = |diff_\rho(\mathcal{T}_Q, \mathcal{T}_D)|$$

Documents are ranked with respect to the dissimilarity coefficients between their descriptions and the query description. The more a document covers the query, the best is its rank.

Example 8. Let us now illustrate the ranking process on the terminologies $\mathcal{T}_Q$, $\mathcal{T}_{D_1}$ and $\mathcal{T}_{D_2}$ of Example 2. The differences between the terminologies are the following

$$diff_\rho(\mathcal{T}_Q, \mathcal{T}_{D_1}) = \text{Comfortable} \sqcap \exists \text{provided-with.}(\text{Bed} \sqcap \exists \text{has-size.Large})$$
$$diff_\rho(\mathcal{T}_Q, \mathcal{T}_{D_2}) = \text{Air conditioned}$$

We can deduce that the second document matches better the query than the first since its difference is the smallest one. Hence, we have the intended result described in Section 2.

6 Discussion

Nowadays, search engines sort their results according to number of criteria going from the number, proximity and location of terms matched, to pages related factors such as the number of links made to a page or the number of times a page is accessed from a results list. The ranking algorithms used by the search engines are not published and we know only a little about their ranking criteria [9]. The novelty of the approach proposed in this paper is that it allows the user to express his query as a natural language description. The criteria used when sorting the retrieved documents is their semantic relevancy w.r.t the query needs.

In the semantic web framework, an approach of ranking query results is proposed in the SEAL semantic portal [13]. Query results are reinterpreted as F-Logic knowledge bases. The semantic ranking is reduced to the comparison of two knowledge bases. A similarity is computed between the query and the knowledge bases, it serves as a basis for the semantic ranking. Their notion of similarity between two terminologies is reduced to the similarity between concept pairs.

Number of similarity measures for ontological structures were proposed in different domains like databases, artificial intelligence and semantic web [12,3, 5]. The work in [12] extends the comparison to semantic structures (set of super and sub-concepts of a concept) and relations between the concepts.

Our approach in matching terminologies is more complete since it operates in semantic descriptions expressed in description logics rather than structures. In addition, it involves both name and complex description matching.

Our name similarity measure is based on Wordnet. Different measures between lexicalized concepts in the Wordnet hierarchy have been proposed (see [4] for a survey). We choose to use the measure proposed by Hirst and St-Onge [7] because it uses all relations in Wordnet, the other measures are based only on hyponymy.

The constant and threshold values proposed in example 7 for computing the mapping between two terminologies are the typical values we have used in our experiments. Those values are subjective and depend on the intended results. For example, for additional restriction of the the mapping, th_{map} can be increased

and allow only for synonyms and direct hypernyms in the set of equivalence names, th_{sim} have to be set to 0.87.

In real-life applications, often exact matching is not realistic. We will investigate in our future work an approximate matching of the query results.

References

1. F. Baader, R. Kusters, and R. Molitor. Computing Least Common Subsumers in Description Logics with Existential Restrictions. In T.Dean, editor, *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI'99)*, pages 96–101. Morgan Kaufmann, 1999.
2. Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge: University Press, 2003.
3. G. Bisson. Learning in FOL with a Similarity Measure. In *10th National Conference on Artificial Intelligence*. Morgan Kaufmann, 1992.
4. A. Budanitsky. Semantic Distance in WordNet: An Experimental, Application-oriented Evaluation of Five Measures, 2001.
5. A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Learning to Map Between Ontologies on the Semantic Web, 2002.
6. C. Fellbaum. *WordNet An Electronic Lexical Database*. The MIT Press, 1998.
7. G. Hirst and D. St-Onge. Lexical Chains as Representation of Context for the Detection and Correction of Malapropisms. In C. Fellbaum, editor, *WordNet: An electronic lexical database and some of its applications. Cambrige, MA: The MIT Press*, 1998.
8. N. Karam and M. Schneider. Comparing Natural Language Documents: a DL Based Approach. In *International Workshop on Description Logics (DL2003)*, 2003.
9. M. Kobayashi and K. Takeda. Information Retrieval on the Web. *ACM Computing Surveys*, 32(2):144–173, 2000.
10. R. Kusters. *Non-Standard Inferences in Description Logics*, volume 2100 of Lecture Notes in Artificial Intelligence. Springer-Verlag, 2001.
11. J. Madhavan, P.A. Bernstein, and E. Rahm. Generic Schema Matching with Cupid. In *The VLDB Journal*, pages 49–58, 2001.
12. A. Maedche and S. Staab. Measuring Similarity between Ontologies. In *European Conference of Knowledge Acquisition and Management - EKAW2002*, Lecture Notes in Computer Science, Madrid, Spain, 2002. Springer.
13. A. Maedche, S. Staab, N. Stojanovic, R. Studer, and Y. Sure. SEAL – A Framework for Developing SEmantic Web PortALs. *Lecture Notes in Computer Science*, 2097, 2001.
14. R. A. Schmidt. Terminological Representation, Natural Language & Relation Algebra. In H. J. Ohlbach, editor, *Proceedings of the sixteenth German AI Conference (GWAI-92)*, volume 671 of *Lecture Notes in Artificial Intelligence*, pages 357–371, Berlin, 1993. Springer.
15. G. Teege. Making the Difference: A Subtraction Operation for Description Logics. In Jon Doyle, Erik Sandewall, and Pietro Torasso, editors, *KR'94: Principles of Knowledge Representation and Reasoning*, pages 540–550. Morgan Kaufmann, San Francisco, California, 1994.