ISTITUTO DI ANALISI DEI SISTEMI ED INFORMATICA
"Antonio Ruberti"
CONSIGLIO NAZIONALE DELLE RICERCHE

Anna Formica and Michele Missikoff

CONCEPT SIMILARITY IN *SYMONTOS*: AN ENTERPRISE ONTOLOGY MANAGEMENT TOOL

R. 554   Settembre 2001

Anna Formica, Michele Missikoff – IASI-CNR, Viale Manzoni 30, I-00185, Rome, Italy
ph.: (+39 06) 77161, fax: (+39 06) 7716461
{formica,missikoff}@iasi.rm.cnr.it.

ISSN: 1128–3378

Collana dei Rapporti dell'Istituto di Analisi dei Sistemi ed Informatica "Antonio Ruberti", CNR

viale Manzoni 30, 00185 ROMA, Italy

tel. ++39-06-77161
fax ++39-06-7716461
email: iasi@iasi.rm.cnr.it
URL: http://www.iasi.rm.cnr.it

Abstract

The possibility of assessing the similarity between concepts is growing in importance. Among the primary reasons, there is the development of the *Net Economy* that requires a high level of computer support and flexibility in doing business. In business transactions, *similarity* plays an important role. It is constantly used whenever a certain good or service is not available with the required characteristics. Then a substitute may be accepted, as far as it is sufficiently close to what was originally required. In this paper we propose a method for evaluating concept similarity. The work has been performed within the *SymOntos* project concerning the development of a symbolic ontology management system, where concepts are defined in accordance with a frame-oriented approach.

Keywords: *Enterprise Ontologies, Concept Similarity, Symbolic Ontology Management Systems.*

1. Introduction

The possibility of assessing the similarity between concepts is growing in importance. Among the primary reasons, we may cite the development of the so called *Net Economy*, that requires flexibility in doing business and the possibility of co-operation for national and international organizations, creating unplanned, often temporary, partnerships. In business transactions, similarity plays an important role. Flexibility requires enterprises to be able to cope with (often unexpected) different situations with respect to what originally planned. For instance, in an e-procurement transaction may be the case that the required good is not available with the desired characteristics (e.g., with the expected price, quality, or delivery date), therefore the production plan must be adjusted to use a ”similar” part, although not exactly the one originally planned. (If the new part is ”very similar”, the production plans do not need to be adjusted.) A similarity evaluation method is also required in different areas, such as ontology integration, integration of multiple heterogeneous information sources for mediation and data warehousing, virtual enterprises, component-based information systems development. It is also important in another, very different, context, such as tourism services. When you start planning an holiday, it is very difficult to find exactly what you are looking for. Often, it is necessary to accept an hotel somehow close to the original choice (but not exactly), a flight with different dates or price. Again, similarity reasoning appears to be a fundamental activity, although we often establish a similarity threshold, below which we simply decide to stop since the trip is no more what we originally wanted.

On a more general ground, *similarity reasoning*, like *taxonomic reasoning* [3], represents one of the key mechanisms that humans use in order to organize their thoughts and plan their actions. However, similarity is a notion very difficult to be precisely and exhaustively defined. Objects can be similar from certain points of view and very different from others. According to [20], if we consider (the notion of) a *pig*, a *donkey*, and a *car*, the first two exhibit a greater affinity being both animals but, in another perspective, the last two are similar as vehicles. The first similarity is due to a natural affinity, the second to a functional affinity. In this paper we consider concept similarity from an informational point of view. Given two concepts, e.g. *car* and *truck*, with their respective definitions, we would like to have a method to assess their similarity. In e-commerce, the e-procurement is performed automatically by machines, then a similarity reasoning facility would be extremely useful in performing automatic transactions [30]. The work presented in this paper is a first solution that has been adopted in *SymOntos* [29], an enterprise ontology management system developed at LEKS (Lab for Enterprise Knowledge and Systems), IASI-CNR, within two European projects, namely *FETISH* (*F*ederated *E*uropean *T*ourism *I*nformation *S*ystem *H*armonization) and, currently, *Harmonise*. *SymOntos* is based on the *OPAL* (*O*bject, *P*rocess, *A*ctor *L*anguage) methodology [15], that allows concepts to be defined according to a frame-oriented approach. Notice that Frame Theory is a paradigm for representing real world knowledge, originally introduced by Minsky in [25], from which numerous research tracks on intelligent systems originated, such as *Natural Languages* and *Recognition* [10], *Hybrid Systems* [6], *Object-Oriented Languages* [21], *ISA-hierarchies* and *subsumption* [7], *F-Logic* [22].

1.1. The Knowledge Representation Method

According to OPAL, an ontology is constructed by defining a set of concepts and establishing semantic relations among them. OPAL supplies a set of predefined concept categories (referred to as *metaconcepts*) and semantic relations that form the OPAL framework. The definition of a domain concept takes place by filling a concept template (conceived according to a frame-slot approach), supplying first the OPAL category it belongs to, then filling of the specified slots. The OPAL concept categories on which we focus in this paper are: *Actor*, *Object*, and *Process*[1].

- *Actor*: this metaconcept allows the ontology engineer to define the active concepts of the domain (e.g., *Customer* or *Travel_Agency*). A concept of this category is able to activate or perform one or more processes;
- *Object*: this metaconcept is used to model passive concepts, on which processes operate (e.g., *Flight_seat*, *Hotel_room*);
- *Process*: this metaconcept is used to model activities, that are performed to achieve actors' goals (e.g., *Hotel_room_reserving*, *Flight_booking*).

Therefore, according to OPAL, a *SymOntos* concept is defined by specifying, besides the *label* and a *description* (d) in natural language, the following slots:

Kind (k) - that specifies the category of the concept being defined (i.e., *Actor*, *Object*, or *Process*);
Broader (B) - that gathers a set of references to more general concepts;
Part (Pa) - that gathers a set of references to concepts representing components;
Related (R) - that gathers a set of references to related concepts;
Predicate (Pr) - that gathers a set of references to concepts that can be seen as attributes;
Similar (S) - that gathers a (possibly empty) set of terms that represent similar concepts. Each term is associated with a *similarity degree* (a positive decimal less or equal to 1.0. In the latter case we have a synonym).

Example 1.1. Below a *SymOntos* concept is shown, whose label is $GuestHouse$, that is defined as follows:
$GuestHouse$:= (
d = "Private house where accommodation and in most cases breakfast are provided",
$k = Object$,
$B = \{Accommodation\}$,
$Pa = \{DiningRoom\}$,
$R = \{Customer, Breakfast\}$,
$Pr = \{Price\}$,
$S = \{\langle Hotel, 0.7\rangle\}$
) □

It is important to note that the *similarity degree* is not judged by the user but it is established, by means of a *Consensus System* [26], by a panel of experts in a preliminary phase. We will refer to it as *tentative similarity* (*tsim*), to distinguish it from *concept similarity* (*csim*), that

[1] Notice that, the examples provided in this paper have been taken from the tourism domain. In particular, with regard to the descriptions of the tourism concepts, we considered the work developed in [13], within the *FETISH* European Project.

is evaluated on the basis of the concept structure and is only partially influenced by *tsim*. The above concept structure allows a complex semantic net [6] to be defined. A few interesting subgraphs can be identified. One is the *inheritance hierarchy*, constructed by means of the *Broader* declarations; another is the *similarity graph*, constructed by means of the *Similar* declarations. The remaining sections of a concept definition (i.e., *Part*, *Related*, and *Predicate*) represent the structural form, since they determine the information structure of the related instances. The aim of this work is to use (i) the inheritance hierarchy, (ii) the similarity graph, and (iii) the concept structural forms to derive the concept similarity *csim*.

1.2. The Essence of the Proposed Method

The proposed method is divided in two phases. The first is a preparation phase, where the concepts are pre-elaborated, in order to make their structures fully explicit. The second is the evaluation phase, where concept similarity is actually computed.

Phase 1 - Expanding the ontology
In this phase, the goal is to analyze the concept definitions to build two graphs. The first is the *inheritance graph* (indeed, a *Directed Acyclic Graph - DAG* - if it is correctly defined), built starting from the *Broader* section of the concept definitions, that organizes the ontology concepts according to a generalization hierarchy. In this phase, the inheritance process is performed, therefore the structural sections of concept definitions (i.e., *Part*, *Related*, and *Predicate*) are augmented with the concept labels inherited from more general concepts (for a full treatment of structural inheritance, please refer to [2]). This operation is referred to as "expansion".

The second is the *similarity graph*, built starting from the *Similar* slot, where nodes are concepts and arcs are labeled with their tentative similarity degree. Since similarity enjoys the reflexive, symmetric, and transitive properties, the *similarity graph* is obtained starting from the original definitions (referred to as *signature for similarity*) and operating the reflexive, symmetric, and transitive closure.

The output of this phase is an expanded (i.e., all the definitions have been expanded exploiting inheritance) set of concepts and two graphs: *inheritance DAG* and *similarity graph.*

Phase 2 - Deriving concept similarity
Starting from the ontology transformed according to the previous phase, concept similarity is evaluated by using their expanded structure. In our approach we consider four notions of similarity. The first is the tentative similarity (*tsim*) declared in the concept definition. Then, we have the following:

- *Flat structural similarity* (*fss*) - This is computed by analyzing the three structural slots (*Pa*,*R*,*Pr*) and evaluating the similarity of every concept referred therein.
- *Hierarchical structural similarity* (*hss*) - This sort of similarity only pertains to concept pairs that are hierarchically related. It is computed starting from the flat structural similarity (*fss*) defined above, by taking into consideration a further element related to the hierarchical relationship. In particular a factor, that represents the probability for an instance of the more general concept to be also an instance of the specialized concept, is introduced.
- *Concept similarity* (*csim*) - This is the final figure that is produced by combining the

structural similarity (either $flat$ or *hierarchical*, depending on the case) and the tentative similarity supplied in the original concept definition.

The rest of this paper is organized as follows. In Section 2 the preliminary definitions of *SymOntos* concept and ontology, with the related notions of structural forms, are given. This allows us to formally address, in Section 3, Phase 1 by defining the structures (essentially, the ontology in expanded form, the *inheritance DAG*, and the *similarity graph*) on which the similarity evaluation analysis is performed. In Section 4 the actual method is described, with the steps of Phase 2 that allow the three mentioned kinds of similarities (fss, *hss*, *csim*) to be derived. Successively, the Related Work Section is given, followed by Section 6, where the conclusion and future lines of research are mentioned.

2. Formal Basis

In *SymOntos*, the fundamental modeling notion is that of a *concept*, specified by a *concept expression.* A concept expression has a left hand side, that is the identifying *label* of the concept (essentially, its name), and a right hand side, the *concept definition*, that specifies the *structure* of the concept. For instance, *Hotel* and *Customer* are concept labels. Below, the notion of a *SymOntos concept* is formally introduced.

Definition 2.1. [SymOntos concept] A *SymOntos concept* (*concept* for short) is a concept expression:

$c := (d,k,B,Pa,R,Pr,S)$

where the left hand side, i.e. c, is a *label* that uniquely identifies the concept, whereas the right hand side, that is the $concept\ definition$, is a 7-tuple defined as follows:

- d is a string expressing the *description* (i.e., the intuitive meaning) of the concept name, in natural language;
- k is the *kind* of the concept (i.e., its category, such as *Actor*, *Object*, or *Process*);
- B is the set of the names of the *Broader* concepts of c, i.e., labels denoting generalizations of c;
- Pa is the set of the names of the concepts that represent components (*Part*) of c;
- R is the set of the names of the concepts that are somehow *Related* to c;
- Pr is the set of the names of the concepts that in *Predicate* relation with c, i.e., denoting attributes of the concept being defined;
- S is the set of pairs $\langle b, tsim\rangle$ where b is the name of a concept that is *Similar* to c, and *tsim* is a decimal number in the interval [0.0 ... 1.0] standing for the *tentative similarity degree.*

Notice that, in the cases where confusion may arise, the components of the 7-tuple and the similarity degree will be properly indexed with the names of the related concept. For instance, the k element will be marked as k_c, and the similarity degree between the concepts c and b will indicated as $tsim_{c,b}$. □

We present now the notion of a *SymOntos ontology.*

Definition 2.2. **[SymOntos ontology]** A *SymOntos ontology* (*ontology* for short) O is a set of interrelated *SymOntos* concepts. In particular, if T_O is the set of all the concept labels appearing in O, then it is partitioned by the sets N_O and W_O, i.e.:
$T_O = N_O \cup W_O$
$N_O \cap W_O = \emptyset$
where N_O is the set of concept labels that are left hand sides of concept expressions in O, and W_O is the set of all the remaining terms appearing in O that are referred to as *known words* of the ontology. □

Known words represent ”boundary concepts” that are intentionally left undefined, i.e., they denote concepts that do not belong to the application domain that is modeled, but are used in some definitions.

A concept label is referred to as a *reference* when it is defined in the right hand side of a concept expression, that is, it is used in a concept definition.

Example 2.1. Consider the concept *GuestHouse* previously defined, together with the following two concepts:

- *Accommodation* := (
  d = ”A place where at least sleeping and sanitary facilities are provided”,
  k = *Object*,
  $B = \{\}$,
  $Pa = \{Room\}$,
  $R = \{Country\}$,
  $Pr = \{NofRooms\}$,
  $S = \{\langle Hotel, 0.8\rangle\}$
)

- *RuralHouse* := (
  d = ”GuestHouse in the countryside”,
  k = *Object*,
  $B = \{GuestHouse\}$,
  $Pa = \{Court\}$,
  $R = \{RusticLand\}$,
  $Pr = \{NofRecrServ\}$,
  $S = \{\ \}$
)

This is a very simple example of ontology where, for instance, *Customer* (in the concept definition of *GuestHouse*) and *Court* (in the concept definition of *RuralHouse*) are known words, i.e., they are not left hand sides of any concept expressions, and *Hotel*, in the *Accommodation* concept definition, is a reference.

□

Indeed, we are not interested in any ontology, rather in the ontologies that satisfy some formal properties, also referred to as *correct* ontologies. Such a notion, that will be formally introduced in Section 3, is based on some properties defined over the structure of the concepts and, in particular, on the mutual references that the concepts of the ontology exhibit.

To this end, below we start by addressing the notion of a *structural form* of a concept, that consists of components (Pa), associations (R), and attributes (Pr) of a concept definition. Since we focus on structural similarity, such a notion is fundamental in computing *concept similarity*, as defined in Section 4.

Definition 2.3. [Structural form of a concept] The *structural form* of a concept c is the concept expression whose name is indicated as c^- and whose definition is given by the three structural slots Pa,R, and Pr of c, i.e.:
$c^- := (Pa,R,Pr)$ □

Example 2.2. For instance, the structural form of the *GuestHouse* concept of the Example 1.1 is defined as follows:
$GuestHouse^- := ($
$Pa = \{DiningRoom\}$,
$R = \{Customer,\ Breakfast\}$,
$Pr = \{Price\}$
$)$ □

In the following, given an ontology O, the set of the structural forms of the concepts defined in O will be denoted as O^-.

The slots of a concept definition that are not present in the structural form, i.e. B and S, are used to define the *signature for inheritance* and *signature for similarity* of the *structural form* of an ontology, respectively, as defined below. Notice that the notion of a signature for inheritance has been originally introduced in [1]. In this paper, such a notion will be used in accordance with [2], where it represents the *DirectDesc* relation (i.e., the relation among a concept and its immediate specializations).

Definition 2.4. [Structural form of an ontology] Given an ontology O, let $\mathcal{O}$ be the triple (O^-, Σ_O, Γ_O), where O^- is the set of structural forms of the concepts in O, and Σ_O, Γ_O are two relations defined as follows:

- Σ_O is a set of ordered pairs defined according to the inheritance hierarchy (the sets of broader concepts) of O as follows:
  $\Sigma_O = \{\langle d, c\rangle \mid c, d \in T_O \text{ and } d \in B_c\}$;
- Γ_O is a set of ordered triples defined according to the sets of similar concepts of O as follows:
  $\Gamma_O = \{\langle c,d,tsim\rangle \mid c, d \in T_O \text{ and } \langle d,tsim\rangle \in S_c\}$.

Then, $\mathcal{O}$ = (O^-, Σ_O, Γ_O) is the *structural form* of the ontology O, where Σ_O and Γ_O are referred to as the *signature* for *inheritance* and *signature* for *similarity* of $\mathcal{O}$, respectively. □

Example 2.3. For instance, suppose to enrich the ontology given by the concepts of the Examples 1.1, 2.1 with the further concepts:

- $FarmHouse := ($
  d = "GuestHouse located on an operating farm",
  $k = Object$,
  $B = \{GuestHouse\}$,

$Pa = \{Dairy\}$,
$R = \{Countryside, Milk, Cheese\}$,
$Pr = \{NofAnimals\}$,
$S = \{\}$
)

- $Hotel :=$ (
  $d =$ "Establishment with reception, services and additional facilities where accommodation and in most cases meals are provided",
  $k = Object$,
  $B = \{Accommodation\}$,
  $Pa = \{Restaurant\}$,
  $R = \{Tourist, CreditCard\}$,
  $Pr = \{Cost, NofCreditCards\}$,
  $S = \{\}$
)

- $GrandHotel :=$ (
  $d =$ "Hotel where accommodation is provided in rooms or suites",
  $k = Object$,
  $B = \{Hotel\}$,
  $Pa = \{Suite, SwimmingPool\}$,
  $R = \{Limousine, Airline\}$,
  $Pr = \{NofSuites, LimoService\}$,
  $S = \{\langle Hotel, 0.9\rangle\}$
).

The structural form of this ontology is given by the structural forms of the concepts in it defined, and the signatures for inheritance and similarity graphically represented in Figures 1, and 2, respectively. Notice that the evaluation of the similarity degrees between, for instance, *GuestHouse* and *GrandHotel*, or *GrandHotel* and *Accommodation*, will be addressed in the next section. □

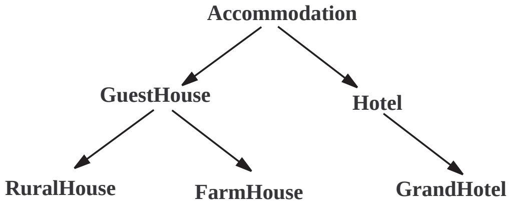

Figure 1: Signature for inheritance of the Example 2.3

3. Correct Ontology

In this section the conditions that an ontology has to satisfy to be *correct* are presented. As we will see, the notion of a correct ontology concerns the signatures for inheritance and similarity

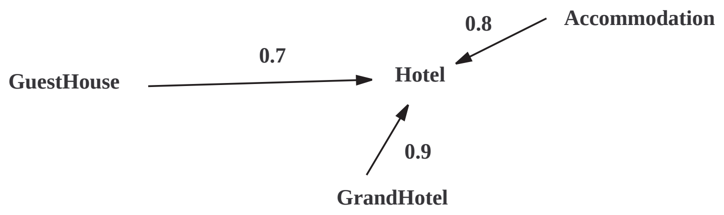

Figure 2: Signature for similarity of the Example 2.3

of such an ontology. In order to address the formal properties regarding the signature for similarity, in the following subsection some definitions about terms similarity (i.e., tentative similarity between concept labels) are first introduced.

3.1. Formal Properties about Term Similarity

In this subsection a few definitions concerning the similarity among a general set of terms (i.e., concept labels, defined or undefined) T is given.

In the following, for sake of simplicity, given two concepts names c_i, c_j, the tentative similarity degree $tsim_{c_i,c_j}$ will be indicated as $tsim_{i,j}$.

Definition 3.1. [Tentative similarity] Given a set of terms T, the *tentative similarity* (*similarity* for short) is a relation on $T \times T \times [0.0...1.0]$ where, if $\langle c_i, c_j, tsim_{i,j} \rangle \in T \times T \times [0.0...1.0]$, the decimal number $tsim_{i,j}$ is referred to as the *tentative similarity degree.* □

Below, the notions of $reflexive$, *symmetric*, and *transitive* similarity are given, together with their *closures*.

Definition 3.2. [Reflexive Similarity] A similarity S on $T \times T \times [0.0...1.0]$ is $reflexive$ if and only if:

$\forall\ c_i \in T \Rightarrow \langle c_i, c_i, tsim_{i,i} \rangle \in T$ and $tsim_{i,i} = 1.0$.

Furthermore, given two similarity relations S and R on $T \times T \times [0.0...1.0]$, S is the *reflexive similarity closure* of R if and only if S is the smallest subset of $T \times T \times [0.0...1.0]$ such that:

- S contains R;
- S is reflexive. □

Of course, S is obtained from R by adding all the elements $\langle c_i, c_i, 1.0 \rangle$, for all $c_i \in T$.

Definition 3.3. [Symmetric similarity] A similarity S on $T \times T \times [0.0...1.0]$ is *symmetric* if and only if:

$\forall\ \langle c_i, c_j, tsim_{i,j} \rangle \in S \Rightarrow \langle c_j, c_i, tsim_{j,i} \rangle \in S$ and $tsim_{i,j} = tsim_{j,i}$.

Furthermore, given two similarity relations S and R on $T \times T \times [0.0...1.0]$, S is the *symmetric similarity closure* of R if and only if S is the smallest subset of $T \times T \times [0.0...1.0]$ such that:

- S contains R;
- S is symmetric. □

Of course, S is obtained from R by adding all the elements $\langle c_j, c_i, tsim_{i,j} \rangle$, for all $\langle c_i, c_j, tsim_{i,j} \rangle$ in R.

Definition 3.4. [Transitive similarity] A similarity S on $T{\times}T{\times}[0.0...1.0]$ is *transitive* if and only if:

$\forall\ \langle c_i, c_j, tsim_{i,j}\rangle,\ \langle c_j, c_h, tsim_{j,h}\rangle \in S \Rightarrow \langle c_i, c_h, tsim_{i,h}\rangle \in S$

where $tsim_{i,h}$ is a value depending on $tsim_{i,j}$, and $tsim_{j,h}$, i.e.:

$tsim_{i,h} = f(tsim_{i,j}, tsim_{j,h})$

such that $tsim_{i,h} \leq tsim_{i,j}$,$tsim_{j,h}$.

Furthermore, given two similarity relations S and R on $T{\times}T{\times}[0.0...1.0]$, S is the *transitive similarity closure* of R if and only if S is the smallest subset of $T{\times}T{\times}[0.0...1.0]$ such that:

- S contains R;
- S is transitive. □

Of course, S is obtained from R by adding all the elements $\langle c_i, c_h, tsim_{i,h}\rangle$, for all $\langle c_i, c_j, tsim_{i,j}\rangle$, $\langle c_j, c_h, tsim_{j,h}\rangle$ in R.

Notice that the above definition has been conceived in order to give maximum flexibility to the method. In fact, the f function can be defined by the user according to the specific application domain addressed. For instance, in this paper, we assume that:

$f(tsim_{i,j}, tsim_{j,h}) = tsim_{i,j} * tsim_{j,h}$

but more sophisticated choices are compatible with the method, e.g., "fuzzy functions".

Example 3.1. For instance, in our example, if we assume that $T = T_O$, let R be the similarity represented in Figure 2, i.e.:

$\langle GuestHouse, Hotel, 0.7\rangle$,
$\langle GrandHotel, Hotel, 0.9\rangle$,
$\langle Accommodation, Hotel, 0.8\rangle$.

By adding to it the following triples:

$\langle GuestHouse, GuestHouse, 1.0\rangle$,
$\langle GrandHotel, GrandHotel, 1.0\rangle$,
$\langle FarmHouse, FarmHouse, 1.0\rangle$,
$\langle RuralHouse, RuralHouse, 1.0\rangle$,
... (for all the concepts names defined in T_O).

we have the reflexive similarity closure of R. Furthermore, by adding:

$\langle Hotel, GuestHouse, 0.7\rangle$,
$\langle Hotel, GrandHotel, 0.9\rangle$,
$\langle Hotel, Accommodation, 0.8\rangle$,

we get the symmetric similarity closure of R.

Notice that the transitive similarity closure of R is R itself, since it is not possible to derive triples by transitivity in it. However, if the transitive similarity closure is applied to the symmetric similarity closure of R, it is possible to derive:

$\langle GuestHouse, Accommodation, 0.56\rangle$,
$\langle GrandHotel, Accommodation, 0.72\rangle$,

$\langle GuestHouse, GrandHotel, 0.63\rangle$.

Therefore, in order to obtain all possible triples that can be derived by transitivity, the symmetric similarity closure will be applied first. This is illustrated in the next subsection. □

3.2. Inheritance DAG and Similarity Graph

As already mentioned, the formal definition of a *correct* ontology is related to some formal properties that the signatures for inheritance and similarity of the ontology have to satisfy. To this end, below the notions of *inheritance DAG* and *similarity graph* are first introduced.

Definition 3.5. [Inheritance DAG] Given an ontology O, consider its structural form $\mathcal{O} = (O^-, \Sigma_O, \Gamma_O)$. Let $\Sigma_{\mathcal{O}}$ be the transitive closure of Σ_O. Consider the following conditions:

1. $\Sigma_{\mathcal{O}}$ is antireflexive;
2. $\Sigma_{\mathcal{O}}$ is antisymmetric;
3. $\forall\ \langle c,d\rangle \in \Sigma_{\mathcal{O}} \Rightarrow k_c = k_d$, i.e., the concepts have the same *kind*.

Then, if all the above conditions are fulfilled, $\Sigma_{\mathcal{O}}$ is referred to as the *inheritance DAG* of $\mathcal{O}$. □

In fact, it is well known that the inheritance hierarchy of a set of concepts must be free of cycles and, in particular, the inheritance relation has to be antireflexive, antisymmetric, and transitive, i.e., $(T_O, \Sigma_{\mathcal{O}})$ must be a strict partially ordered set (POSET) [17].

For instance, the transitive closure of the signature for inheritance represented in Figure 1 fulfills all the three conditions given in the previous definition. Then, it is the inheritance DAG of the ontology described in the Example 2.3.

Definition 3.6. [Similarity graph] Given an ontology O, consider its structural form $\mathcal{O} = (O^-, \Sigma_O, \Gamma_O)$. According to the Definition 2.4, Γ_O is a similarity on $T_O \times T_O \times [0.0...1.0]$. Let $\Gamma_{\mathcal{O}}$ be the transitive closure of the reflexive and symmetric closure of Γ_O. Consider the following conditions:

1. $\forall\ \langle c, d, tsim_{c,d}\rangle \in \Gamma_{\mathcal{O}} \Rightarrow k_c = k_d$, i.e., the concepts have the same *Kind*;
2. $\forall\ c,d \in T_O$, $\langle c, d, as_{c,d}\rangle \in \Gamma_{\mathcal{O}}$, where $as_{c,d}$ is defined as follows:
   $as_{c,d} = tsim_{c,d}$ if $tsim_{c,d}$ is the similarity degree defined in $\Gamma_{\mathcal{O}}$ and
   it is unique;
   $as_{c,d} = \{tsim^i_{c,d}\}_{Choice}$ in the presence of multiple (transitively
   derived) similarity degrees defined in $\Gamma_{\mathcal{O}}$;
   $as_{c,d} = 0.0$ otherwise.

If all the above conditions are fulfilled, $\Gamma_{\mathcal{O}}$ is referred to as the *similarity graph* of $\mathcal{O}$. In particular, $as_{c,d}$ will be referred to as the *axiomatic similarity* degree of the concepts c,d. □

For instance, the transitive similarity closure of the symmetric similarity closure of the signature for similarity represented in Figure 2 is shown in Figure 3.

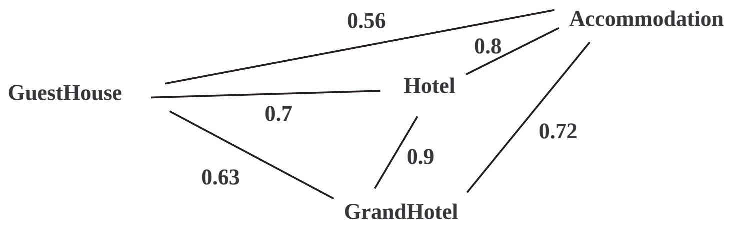

Figure 3: Similarity subgraph of the Example 2.3

Notice that in this case, if we consider the symmetric closure of Figure 2, for each pair of concept names the similarity degrees derived by transitivity are unique. Therefore, by extending the graph of Figure 3 with reflexivity, and the triples:

$\langle GuestHouse, FarmHouse, 0.0\rangle$
$\langle RuralHouse, GrandHotel, 0.0\rangle$
.... (for all pairs not involved in any similarity)

we have the similarity graph of the ontology described in the Example 2.3.

Finally, we have the notion of a *correct SymOntos* ontology.

Definition 3.7. [**Correct ontology**] Given an ontology O, consider its structural form $\mathcal{O} = (O^-, \Sigma_O, \Gamma_O)$. Then, the ontology O is *correct* iff $\Sigma_{\mathcal{O}}$ is the inheritance DAG and $\Gamma_{\mathcal{O}}$ is the similarity graph of $\mathcal{O}$. □

If we extend the similarity subgraph of Figure 3 as mentioned above, the ontology of the Example 2.3 is correct.

3.3. Concepts Inheritance

As already mentioned in the Introduction, the goal of the paper is the definition of a method that allows similarity among ontology concepts to be evaluated on the basis of the concept definitions. In order to perform this evaluation, the "expansion" step must be performed. Such a step concerns the inheritance of the concept definitions, a problem widely investigated in literature, see for instance [2]. The inheritance process is a necessary step for the evaluation of structural similarity, since in the structural form of a concept all the concept labels declared in the slots of its ancestors, up in the inheritance DAG of the ontology, must be present. The inheritance process is performed by applying to the ontology concepts the *Expand* function that will be illustrated in the next subsection. Such a function is a revisitation of the *Expand* function defined in [2], modified in order to deal with the richer knowledge model used in OPAL to construct concept expressions.

Below, given a correct ontology, the notion of *Ancestors* of a concept is introduced. Such a notion allows all the concept names that are generalizations of a given concept, up in the inheritance DAG, to be identified.

Definition 3.8. [**The** *Ancestors* **function**] Consider the structural form $\mathcal{O} = (O^-, \Sigma_O, \Gamma_O)$ of a correct ontology with a non-empty inheritance DAG $\Sigma_{\mathcal{O}}$. Then, the *Ancestors* ($\mathcal{A}$) function is defined as follows:

$\mathcal{A}$: $T_O \rightarrow \wp(T_O)$,

and, given a concept name $c \in T_O$:
$\mathcal{A}(c) = \{d \in T_O \mid \langle c,d \rangle \in \Sigma_{\mathcal{O}}\}$ □

Notice that, for any $c \in T_O$, the set $\mathcal{A}(c)$ is always finite since T_O is finite and $\Sigma_{\mathcal{O}}$ is a DAG. For instance, in our example, we have:
$\mathcal{A}(RuralHouse) = \{GuestHouse,\ Accommodation\}$.

In order to evaluate the similarity among concepts, we have to expand the concept definitions by inheriting all the concept names that are related to the ancestors of the concepts, up in the inheritance hierarchy. To this end, the *Expand* function is presented below. Such a function, essentially, returns a concept whose structural components are defined as the union of the corresponding components of the ancestor concepts.

Definition 3.9. [**The** *Expand* **function**] Consider the structural form $\mathcal{O} = (O^-, \Sigma_O, \Gamma_O)$ of a correct ontology with a non-empty signature for inheritance. Let $\mathcal{C}_e$ be the set of all possible concept expressions. Then, the *Expand* ($\mathcal{E}$) function is defined as follows:
$\mathcal{E}$: $N_O \rightarrow \mathcal{C}_e$,
and, given a concept name $c \in N_O$:
$\mathcal{E}(c) = \ c' := (R'_c, Pa'_c, Pr'_c)$
where:
$R'_c = \bigcup_{g \in \mathcal{A}(c)} R_g \cup R_c$;
$Pa'_c = \bigcup_{d \in \mathcal{A}(c)} Pa_d \cup Pa_c$;
$Pr'_c = \bigcup_{e \in \mathcal{A}(c)} Pr_e \cup Pr_c$. □

Notice that, in the above definition, for a known word $w \in W_O$, the set R_w, Pa_w, and Pr_w, are assumed to be empty since they are not concept names known to the ontology.

By using the $\mathcal{E}$ function, we are able to present the notion of $expanded\ form$ of an ontology. Such a form allows us to present, in the next sections, the method for similarity evaluations.

Definition 3.10. [**Expanded form of an ontology**] Given an ontology in structural form $\mathcal{O}$ = $(O^-, \Sigma_O, \Gamma_O)$, let O' be defined as follows:
$O' = \bigcup_{c_i \in N_O} \mathcal{E}(c_i)$
where $\mathcal{E}$ is the Expand function defined above. Then, the triple:
$\mathcal{O}' = (O', \Sigma_O, \Gamma_O)$
is the *expanded* form of the ontology $\mathcal{O}$.

□

In essence, the expanded form is composed of two signatures (for inheritance and similarity) and the set of concepts in expanded structural form.

Example 3.2. Consider again the Examples 2.1, 2.2. The Expand function applied to the concepts *GuestHouse* and *RuralHouse* returns:

- $GuestHouse' := ($
  $Pa = \{Room,\ DiningRoom\}$,
  $R = \{Country,\ Customer,\ Breakfast\}$,
  $Pr = \{NofRooms,\ Price\}$
  $)$

- $RuralHouse' := ($
  $Pa = \{Room, DiningRoom, Court\},$
  $R = \{Country, Customer, Breakfast, RusticLand\},$
  $Pr = \{NofRooms, Price, NofRecrServ\}$
  $)$

□

In the following subsections, the three notions of structural similarity are addressed. In all the three cases, they are defined for concepts of a correct ontology in expanded form.

4. Deriving Concept Similarity

As pointed out in the Introduction, the goal of the paper is the definition of a method that allows similarity among ontology concepts to be derived on the basis of their definitions. In this approach, the following three kinds of similarity evaluation are proposed, depending on the definitions of concepts to be compared:

- *Flat structural similarity* degree, for concepts that are *not* hierarchically related;
- *Hierarchical structural similarity* degree, for concepts that are hierarchically related;
- *Concept similarity* degree, that represents the final concept similarity evaluation, obtained by composing the tentative (axiomatic) similarity and the derived similarity, flat or hierarchical.

We will show that the axiomatic similarity (as) degree, introduced with the similarity graph (see Definition 3.6), plays a fundamental role in all the three kinds of evaluations, not only in the last one.

4.1. Flat Structural Similarity degree

The *flat structural similarity* (fss) degree is computed on the basis of the expanded structural forms of the concepts and the axiomatic similarity degree defined according to the similarity graph. The method presented in this subsection has been inspired to the *maximum weighted matching* problem in bipartite graphs, that can be solved in polynomial time [16]. Informally, it is illustrated as follows.

Consider two concepts whose names are c_i, and c_j, and one of the three slots of the their structural form, for instance *Part* (Pa). Then:

- consider the cartesian product $Pa_{c_i} \times Pa_{c_j}$;
- within the above set, consider all the sets of pairs such that no two pairs in the set share an element. Such sets will be referred to as *candidate* sets of pairs. For instance, assume that Pa_{c_i} and Pa_{c_j} represent a set of boys and a set of girls, respectively, a candidate set of pairs defines a possible set of marriages (when polygamy is not allowed) [16];
- for each candidate set of pairs, consider the sum of the axiomatic similarity degrees of the concept pairs in it;
- the candidate set having the maximal among all the computed sums is chosen.

Therefore, for each slot, elements of c_i are paired with elements of c_j in order to give the maximal sum. The fss of the concepts c_i,c_j is then computed starting from the three maximal values determined for each of the slots Pa,R, and Pr, up to a normalization factor.

Definition 4.1. [The set $\mathcal{C}_{\mathcal{R}}$ of candidate sets of pairs] Consider two concepts c_i, c_j of a correct ontology and let $\mathcal{R}$ be one of the three concept slots Pa (*Part*), R (*Related*), or Pr (*Predicate*). Let $n_{\mathcal{R}}$, $m_{\mathcal{R}}$ be the cardinalities of the sets $\mathcal{R}_{c_i}$, $\mathcal{R}_{c_j}$, respectively, i.e. $n_{\mathcal{R}} = |\mathcal{R}_{c_i}|$, $m_{\mathcal{R}} = |\mathcal{R}_{c_j}|$, and suppose that $n_{\mathcal{R}} {\leq} m_{\mathcal{R}}$.

Then, the set $\mathcal{C}_{\mathcal{R}}(c_i, c_j)$ of candidate sets of pairs is defined by all possible sets of $n_{\mathcal{R}}$ pairs of concept names defined as follows:

$$\mathcal{C}_{\mathcal{R}}(c_i, c_j) = \{\ \{\langle a_1, b_1\rangle \ldots \langle a_{n_{\mathcal{R}}}, b_{n_{\mathcal{R}}}\rangle\} \mid a_h \in \mathcal{R}_{c_i},\, b_h \in \mathcal{R}_{c_j},\, \forall\ h = 1 \ldots n_{\mathcal{R}},$$
$$\text{and } a_h \neq a_k,\, b_h \neq b_l,\, \forall\ k,l \neq h\}. \qquad \Box$$

Below the definition of fss between concepts of a given ontology follows.

Definition 4.2. [Flat structural similarity (fss)] Consider a correct ontology in expanded form, $\mathcal{O}' = (O', \Sigma_O, \Gamma_O)$.

Then, the *flat structural similarity* (fss) of two concepts whose names are c_i, $c_j \in N_O$ is defined as follows:

$$fss(c_i, c_j) = \sum_{\mathcal{R}\in\mathcal{S}} \left[\frac{w_{\mathcal{R}}}{m_{\mathcal{R}}} \max_{P\in\mathcal{C}_{\mathcal{R}}(c_i,c_j)} \left(\sum_{\langle a,b\rangle \in P} as(a,b) \right) \right]$$

where $\mathcal{S} = \{Pa,\, R,\, Pr\}$ (i.e., $\mathcal{R}$ stands for one of the three concept slots defining the structural form of a concept), $\mathcal{C}_{\mathcal{R}}(c_i, c_j)$ and $m_{\mathcal{R}}$ are defined as in the previous definition, and $as(a, b)$ is the axiomatic similarity degree of the concept names a,b, as defined according to the similarity graph $\Gamma_{\mathcal{O}}$. Furthermore $w_{\mathcal{R}}$ is a weight such that:

$$\sum_{\mathcal{R}\in\mathcal{S}} w_{\mathcal{R}} \leq 1$$

$$\Box$$

Notice that $fss(c_i, c_j)$ is always a value between zero and one and, given two concepts c_i, c_j, $fss(c_i, c_j) = fss(c_j, c_i)$.

Example 4.1. In order to provide a more complex example, suppose that the signature for similarity of the Example 2.3 has been extended as shown in Figure 4. Consider the expanded concepts of the Example 3.2, together with the following ones:

- $FarmHouse' := ($
  $Pa = \{Room,\ DiningRoom,\ Dairy\}$,
  $R = \{Country,\ Customer,\ Breakfast,\ Countryside,$
  $Milk,\ Cheese\}$,
  $Pr = \{NofRooms,\ Price,\ NofAnimals\}$
  $)$

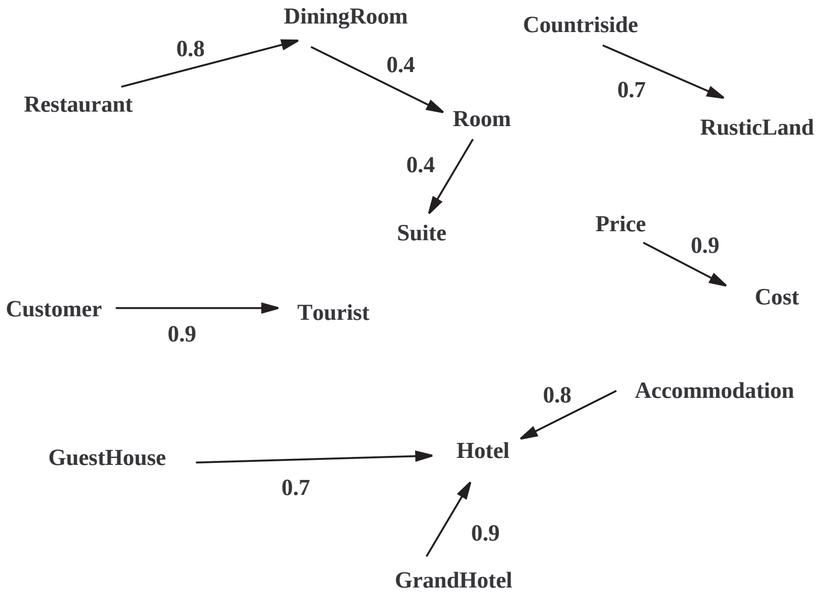

Figure 4: Extended signature for similarity of the Example 2.3

- $GrandHotel' := ($
  $Pa = \{Room,\ Restaurant,\ Suite,\ SwimmingPool\}$,
  $R = \{Country,\ Tourist,\ CreditCard,\ Limousine,$
  $Airline\}$,
  $Pr = \{NofRooms,\ Cost,\ NofCreditCards,\ NofSuites,$
  $LimoService\}$
  $)$

Furthermore for sake of simplicity assume that, for any $\mathcal{R}$, $w_{\mathcal{R}} = \frac{1}{3}$. According to the Definition 4.2, the following holds:

$$\begin{aligned} fss(RuralHouse, FarmHouse) &= \tfrac{1}{3}\ (\tfrac{2}{3} + \tfrac{3{,}7}{6} + \tfrac{2}{3}) = 0.64 \\ fss(RuralHouse, GrandHotel) &= \tfrac{1}{3}\ (\tfrac{1.8}{4} + \tfrac{1.9}{5} + \tfrac{1.9}{5}) = 0.40 \\ fss(FarmHouse, GrandHotel) &= \tfrac{1}{3}\ (\tfrac{1.8}{4} + \tfrac{1.9}{6} + \tfrac{1.9}{5}) = 0.38 \end{aligned}$$

where, for instance, for the concepts $RuralHouse$ and $FarmHouse$ the candidate sets of pairs with maximal sum are the following:

$\{\langle Room, Room\rangle,\ \langle DiningRoom, DiningRoom\rangle,\ \langle Court, Dairy\rangle\}$
$\in \mathcal{C}_{Pa}(RuralHouse, FarmHouse)$
$\{\langle Country, Country\rangle,\ \langle Customer, Customer\rangle,\ \langle Breakfast,\ Breakfast\rangle,$
$\langle Countryside, RusticLand\rangle\} \in \mathcal{C}_R(RuralHouse, FarmHouse)$
$\{\langle NofRooms, NofRooms\rangle,\ \langle Price, Price\rangle,\ \langle NofAnimals, NofRecreServ\rangle\}$
$\in \mathcal{C}_{Pr}(RuralHouse, FarmHouse)$

Intuitively, in order to obtain the maximal sum, it is reasonable to pair the same concept names, leaving the remaining ones to match each other. For instance, in the case of *Related* (R), *RusticLand* has been paired with *Countryside* rather than *Milk* or *Cheese*, since the axiomatic similarity between them is 0.7 rather than 0.0. In the case of *Predicate* (Pr), $NofRecrServ$ has been paired with $NofAnimals$ since, although their axiomatic similarity is 0.0, the sum of the axiomatic similarity degrees obtained from the other two pairs is maximal. □

4.2. Hierarchical Structural Similarity degree

The *hierarchical structural similarity* (hss) degree is computed for concepts that are hierarchically related. The hss is essentially defined as the fss increased by a value defined according to the inheritance DAG of the ontology. In particular, such a value is computed under specific assumptions that are related to the extentional notion of inheritance, i.e., the distribution of concept instances along the hierarchy. This proposal has been formulated under the following assumptions. In the inheritance DAG:

- the concepts are organized according to a *specialization as partition*: in the hierarchy, the instances populate the leaves of the DAG, and the population of an intermediate node is the union of the populations of the children (recursively);
- for any concept, the distribution of the instances among the specialized concepts is uniform, i.e., the children are equally populated.

Such assumptions can be easily relaxed by introducing appropriate coefficients that take into account the actual distribution of instances of the different concepts. Very often, especially in e-business, an ontology is related to a database and, therefore, distribution coefficients can be obtained by means of simple data mining operations (a further elaboration on this point is beyond the scope of this paper). Then, the corrector we propose in order to compute the structural similarity of two hierarchically related concepts is given by the *specialization probability* defined below. It is, essentially, the probability for an instance of a more general concept to be an instance of one of its specialized concepts, under the assumptions above.

Definition 4.3. [Specialization Probability] Consider an inheritance DAG and two concepts c_i,c_j hierarchically related. Let (c_1, ... , c_n) be the path connecting such concepts, where $c_1 = c_i$, that we assume to be more general than c_j, and $c_n = c_j$. Then, if g_h is the outdegree of the concept c_h in the inheritance DAG, for $h = 1...n-1$, the *specialization probability*, say $p(c_i, c_j)$, is defined as follows:

$$p(c_i, c_j) = \prod_h \frac{1}{g_h}$$

□

Then, the hss can be defined as follows.

Definition 4.4. [Hierarchical structural similarity (hss)] Consider a correct ontology in expanded form $\mathcal{O}' = (O', \Sigma_O, \Gamma_O)$, and two concepts c_i,$c_j \in N_O$ that are hierarchically related (i.e., connected by a path) in the signature for inheritance Σ_O. Then, the *hierarchical structural similarity* (hss) of c_i,c_j is defined starting from the flat structural similarity $fss(c_i, c_j)$ as follows:

$$hss(c_i, c_j) = fss(c_i, c_j) + (1 - fss(c_i, c_j)) * p(c_i, c_j)$$

where $p(c_i, c_j)$ is the specialization probability as defined above. □

Example 4.2. For instance, consider the hierarchically related concepts *RuralHouse* and *GuestHouse*. Their flat structural similarity degree is:

$fss(RuralHouse, GuestHouse) = \frac{1}{3}\ (\frac{2}{3} + \frac{3}{4} + \frac{2}{3}) = 0.69$

Now, since the outdegree of the node labeled with *GuestHouse* in the inheritance DAG is 2, then $p(c_i, c_j) = \frac{1}{2}$. Therefore, the hierarchical structural similarity degree between *RuralHouse* and *GuestHouse* is:

$hss(RuralHouse, GuestHouse) = 0.69 + \frac{1-0.69}{2} = 0.84.$ □

4.3. Concept Similarity degree

After the introduction of the fss and the hss degrees, we are able to define the *concept similarity* (*csim*) degree. Is is essentially given by the average of the axiomatic similarity degree as and the hss or fss degrees, if the concepts are hierarchically related or not, respectively.

Definition 4.5. [Concept similarity (csim)] Consider a correct ontology in expanded form, $\mathcal{O}' = (O', \Sigma_O, \Gamma_O)$ and two concepts c_i,$c_j \in N_O$. Then, the *concept similarity* (*csim*) of c_i,c_j is defined as follows. Assume that:
$ss(c_i, c_j) = fss(c_i, c_j)$, if c_i,c_j are not hierarchically related,
$ss(c_i, c_j) = hss(c_i, c_j)$, otherwise.
Then:

$$csim(c_i, c_j) = \frac{ss(c_i, c_j) + as(c_i, c_j)}{2}$$

where $as(c_i, c_j)$ is the axiomatic similarity degree of the concepts c_i,c_j. □

Example 4.3. For instance, in our example, consider the concepts *GuestHouse* and *GrandHotel* that are not related in the inheritance DAG, but they are related in the similarity graph with non-null axiomatic similarity degree. Then:
$csim(GuestHouse, GrandHotel) = \frac{0.40+0.63}{2} = 0.51$
since:
$fss(GuestHouse, GrandHotel) = \frac{1}{3}\ (\frac{1.8}{4} + \frac{1.9}{5} + \frac{1.9}{5}) = 0.40.$
In the case of *RuralHouse* and *FarmHouse* we have two concepts that are again not related in the inheritance DAG, but this time with null axiomatic similarity degree. Therefore:
$csim(RuralHouse, FarmHouse) = \frac{0.64+0.0}{2} = 0.32.$
Consider now the concept *Hotel*, whose expanded form is:

$Hotel' := ($
$Pa = \{Room,\ Restaurant\},$
$R = \{Country,\ Tourist,\ CreditCard\},$
$Pr = \{NofRooms,\ Cost,\ NofCreditCards\}$
$)$

Then, consider *Accommodation* that is hierarchically related to *Hotel*, with non-null axiomatic similarity degree. The following holds:
$csim(Hotel, Accommodation) = \frac{0.69+0.80}{2} = 0.75$
since:
$fss(Hotel, Accommodation) = \frac{1}{3}\ (\frac{1}{2} + \frac{1}{3} + \frac{1}{3}) = 0.38$
$hss(Hotel, Accommodation) = 0.38 + (1 - 0.38)\ \frac{1}{2} = 0.69.$

Finally, as an example of hierarchically related concepts with null axiomatic similarity degree, consider $RuralHouse$ and $GuestHouse$, for which the following holds:
$csim(RuralHouse, GuestHouse) = \frac{0.84+0.0}{2} = 0.42.$ □

5. Related Work

Similarity has been tackled in different fields of *Computer Science*, and a number of significant results are available. Other disciplines, such as *Linguistics* and *Cognitive psychology*, have addressed the same problem producing interesting results, but with a limited impact for us, due to the completely different methodological ground [24]. The method proposed in this paper is the result of the analysis of different solutions that are present in the literature, and our aim is to overcome a number of limitations that we found therein. It must be noted that the large majority of existing results have not been conceived in e-commerce and business-to-business interoperability contexts, but rather in data integration for distributed query processing and/or data warehousing [9, 19, 18, 11]. Therefore what we have perceived as a limitation for our aim may be valid for different applications.

The first difference of the proposed approach with respect to existing results is the way we treat similarity between hierarchically related concepts. For instance, in [9, 14] a constant value (specifically 0.5) is associated with *any* pair of hierarchically related concepts. In our opinion, a constant value does not properly reflect the level of specialization and, on the contrary, it is important to evaluate this coefficient by considering the degree of refinement of the specialized concept: the greater is the refinement, the higher is the distance between the concepts. Therefore, by introducing the notion of hss, we take into account the probability for an instance of a general concept to be also instance of a specialization (e.g., the probability that a *vehicle* is a *car*, in a given application domain). We believe that this method produces better results than merely associating a constant factor to any pair of hierarchically related concepts. For instance, instead of axiomatically assigning 0.5 to the pair of concepts ($Hotel$,*Accommodation*) of the Examples 2.1, and 2.3, three different values can be derived according to the proposed approach:

(i) the first value, $fss(Hotel, Accommodation) = 0.38$, takes into account the structures of the concepts and, in particular, the fact that $Hotel$ has, for each slot, a few of concepts that are not present in the corresponding slots of *Accommodation*;

(ii) the second value, $hss(Hotel, Accommodation) = 0.69$, is obtained by considering the hierarchy of Figure 1 and, in particular, the outdegree of *Accommodation*;

(iii) finally, the average of the previous value with the similarity degree axiomatically given in Figure 4 (in this case 0.8) leads to the final result:
$csim(Hotel, Accommodation) = 0.75$.

Similarity among hierarchically related concepts has been investigated within *Semantic nets* and logic-based *Knowledge Representation.* In [27], where a metric on the power set of nodes in a semantic net has been proposed, the conceptual distance of concepts that are hierarchically related has been defined by considering the length of the shortest path connecting them. Furthermore, in [8] the *Semantic-Distance Metric* (SDM) has been defined, which is based on weighted paths. In particular, in that paper concepts are connected by hyperonym/hyponym and synonym links. With respect to [27], in this paper the hss allows a more refined similarity evaluation that takes into account not only the distance but also the outdegrees of the concepts in the inheritance hierarchy. With respect to [8], in this work not only synonyms have been considered, but also concepts with similarity degrees strictly lesser than one. Furthermore, according to the fss, in our proposal structural links have also been addressed, such as the ones related to attributes or components.

The second main difference of our proposal with respect to the existing literature is the partitioning of the structural definition of a concept into different slots - essentially, attributes (Pr), parts (Pa), and related (R) concepts - comparing therefore only elements of concept definitions that belong to the same partitions. Conversely, the majority of methods found in the literature consider one kind of slot only, namely property names (i.e., attributes). In particular, in our approach these three slots are addressed separately (since the relationship of a *car* with the attribute *colour* is inherently different from its relationship with a *garage* where it is repaired).

In [28], a richer set of distinguishing characteristics has been proposed, that includes both the intentional (classes) and extentional (tokens) levels. However, there are a number of limitations, such as the necessity that two concepts are at the same ISA level to be compared.

On a more technical ground, we did not adopt the popular *Dice*'s function [23], as for instance in [4, 9], that allows concept similarity to be evaluated on the basis of the number of similar concept components divided by the total number of concept components of the two concepts, without explicitly considering in the computation their similarity degree. Therefore, with respect to our approach, such a function introduces a simplification since each similar component counts one, independently of the similarity degree. Analogously, in [12] semantic relatedness (similarity) evaluation is based on the aggregation of the interconnections between concepts, that is, the more properties two concepts have in common, the more closely related they are.

Finally, it is worth mentioning that the fss evaluation between concepts defined in this paper can be seen as a form of *co-occurrence* strategy as defined in [24], for which a *SymOntos* concept is a context and similarity is established on the basis of the amount of overlap of the contexts. Furthermore, in [5], general forms of distance metrics for the computation of similarity measures have been defined, although with more emphasis on the evaluation of similarity between instances, rather than concepts.

6. Conclusion

In this paper a method for the evaluation of concept similarity has been presented. The problem of concept similarity is a complex one, therefore we addressed it from a specific angle: that of structural similarity. Structural similarity, though being a partial view of a more general problem, represents an important issue in the emerging applications of e-commerce. In fact, the structural aspect of a concept determines the structure of data that commercial institutions exchange in doing business. Another field where structural similarity is relevant is that of information integration in query processing of heterogeneous data sources and data warehousing. Even if we consider the structural components of concepts only, the problem appears quite complex. For this reason, in the paper we did not elaborate on a number of tuning parameters, such as the specialization probability factor.

The similarity evaluation method proposed in this paper has been included in the *SymOntos* system [29], developed within the European projects $FETISH$ and *Harmonise*, aiming at the construction and maintenance of tourism ontologies. The method will be used within various tasks, such as semantic data reconciliation and approximate query processing.

References

[1] Ait-Kaci, H. and Podelski, A. (1993) Towards a Meaning of Life. J. of Logic Programming, 16, 195-234.

[2] Beeri, C. and Formica, A. and Missikoff, M. (1999) Inheritance Hierarchy Design in Object-Oriented Databases. Data & Knowledge Engineering, 30(3), 191-216.

[3] Bergamaschi, S. and Sartori, C. (1992) On Taxonomic Reasoning in Conceptual Design. ACM Transactions on Database Systems, 17(3), 385-422.

[4] Bergamaschi, S. and Castano, S. and De Capitani di Vimercati, S. and Montanari, S. and Vicini, M. (1998) An Intelligent Approach to Information Integration. In Guarino, N. (ed), Formal Ontology in Information Systems. IOS Press, Amsterdam.

[5] Bisson, G. (1992) Learning in FOL with a similarity measure. Proceedings of 10th National Conference on Artificial Intelligence, San Jose, CA, July 12-16, 82-87, The AAAI Press/The MIT Press, CA.

[6] Brachman, R.J. (1979) On the epistemological status of semantic networks. In Findler, N.V. (ed), AssociativeNetworks - Representation and use of Knowledge by Computers. Academic Press, New York.

[7] Brachman, R.J. (1983) What IS-A Is and Isn't: An Analysis of Taxonomic Links in Semantic Networks. IEEE Computer, 16(10), 30-36.

[8] Bright, M. and Hurson, A. and Pakzad, S. (1994) Automated Resolution of Semantic Heterogeneity in Multidatabases. ACM Transactions on Database Systems, 19(2), 212-253.

[9] Castano, S. and De Antonellis, V. and Fugini, M.G. and Pernici, B. (1998) Conceptual Schema Analysis: Techniques and Applications. ACM Transactions on Database Systems, 23(3), 286-332.

[10] Charniak, E. (1981) A common representation for problem solving and language comprehension information. Artificial Intelligence, 16, 225-255.

[11] Cohen, W.W. (2000) Data Integration Using Similarity Joins and a Word-Based Information Representation Language. ACM Transactions on Information Systems, 18(3), 288-321.

[12] Collins, A. and Loftus, E. (1975) A Spreading Activiation Theory on Semantic Processing. Psychological Review, 82, 407-428.

[13] Comité Europeen de Normalisation (CEN) (2000) Tourism services - Hotel and other types of tourism accommodation - Terminology. http://www.cenorm.be/

[14] Damiani, E. and Formica, A. and Fugini, M.G. and Missikoff, M. and Pizzicannella, R. (1997) Reusing Analysis Schemas in ODB Applications: a Chart Based Approach. First East-European Symposium on Advances in Databases and Information Systems, St.Petersburg, Russia, September 2-5, 406-415, Nevsky Dialect, Russia.

[15] Formica, A. and Missikoff, M. (2000) Design and specification of an integrated knowledge base. First guidelines to the tourism organisations for enhancing their interoperability in doing business by using a knowledge-based software environment. Deliverable D1.2 of the European Project IST 13015 - FETISH (Federated European Tourism Information System Harmonization), IASI-CNR, Rome, Italy.

[16] Galil, Z. (1986) Efficient algorithms for finding maximum matching in graphs. ACM Computing Surveys, 18, 23-38.

[17] Horowitz, E. and Sahni, S. (1983) Fundamentals of Computer Algorithms. Computer Science Press, Maryland.

[18] Inmon, W.H. (1996) Building the Data Warehouses. John Wiley&Sons, New York.

[19] Jarke, M. and Lenzerini, M. and Vassiliou, Y. (1999) Fundamentals of Data Warehouses. Springer-Verlag, Berlin.

[20] Kasahara, K. and Matsuzawa, K. and Ishikawa, T. and Kawaoka, T. (1995) Viewpoint-Based Measurement of Semantic Similarity between Words. Proceedings of the Fifth International Workshop on Artificial Intelligence and Statistics, Fort Lauderdale, FL, January 4-7, 292-302.

[21] Khoshafian, S. and Abnous, R. (1990) Object-Orientation - Concepts, Languages, Databases, User Interfaces. Wiley, New York.

[22] Kifer, M. and Lausen, G. (1989) F-Logic: A Higher-Order Languge for Reasoning about Objects, Inheritance, and scheme. Proceedings of ACM SIGMOD International Conference on Management of Data, Portland, Oregon, May 31 - June 2, 134-146.

[23] Maarek, Y.S. and Berry, D.M. and Kaiser, G.E. (1991) An Information Retrieval Approach for Automatically Constructing Software Libraries. IEEE Transactions on Software Engineering, 17(8), 800-813.

[24] Miller, G.A. and Charles, W.G. (1991) Contextual correlate of semantic similarity. Language and Cognitive Processes, 6(1), 1-28.

[25] Minsky, M. (1974) A Framework for Representing Knowledge. Artificial Intelligence Memo 306, MIT AI Lab.

[26] Missikoff, M. and Wang, X.F. (2001) A group decision system for collaborative ontology building. Proceedings of Int'l Conference on Group Decision and Negociation, La Rochelle, France, June 4-7, 153-160.

[27] Rada, R. and Mili, H. and Bicknell, E. and Blettner, M. (1989) *Development and application of a metric on semantic nets*; IEEE Transactions on Systems, Man, and Cybernetics, 19(1), 17-30.

[28] Spanoudakis, G. and Constantopoulos, P. (1994) Similarity for Analogical Software Reuse: A Computational Model. Proceedings of the Eleventh European Conference on Artificial Intelligence, Amsterdam, The Netherlands, August 8-12, 18-22, John Wiley&Sons, New York.

[29] *SymOntos: an enterprise ontology management system*; IASI-CNR, www.symontos.org.

[30] Uschold, M. and King, M. and Moralee, S. and Zorgios, Y. (1998) The Enterprise Ontology. The Knowledge Engineering Review, 13(1), 31-89.