Journal of Web Engineering, Vol. 12, No.3&4 (2013) 215-231

A CONCEPTUAL GRAPH BASED APPROACH FOR MAPPINGS AMONG MULTIPLE FUZZY ONTOLOGIES

LINGYU ZHANG

College of Information Science and Engineering, Northeastern University

Shenyang 110819 P.R. China

LI YAN

School of Software, Northeastern University

Shenyang 110819 P.R. China

yanl@swc.neu. edu.cn

Z. M. MA

College of Information Science and Engineering, Northeastern University

Shenyang 110819 P.R. China

Received August 16, 2012
Revised December 25, 2012

Fuzzy ontology mapping is an important tool to solve the problem of interoperation among heterogeneous ontologies containing fuzzy information. At present, some researches have been done to expand existing mapping methods to deal with fuzzy ontology. However, these methods can not perform well when creating mappings among multiple fuzzy ontologies in a specific domain. To this end, this paper proposes a new method for fuzzy ontology mapping called FOM-CG (Fuzzy Ontology Mapping based on Conceptual Graph). To reduce unnecessary comparisons for multiple fuzzy ontologies in a domain, FOM-CG firstly creates or finds out a Reference Ontology that contains the most common and shared information. The other fuzzy ontologies in the domain are Source Ontologies. Then, these fuzzy ontologies are transformed into conceptual graph sets (i.e. R-set and S-sets). Next, some algorithms are presented to create mappings among conceptual graph sets. Finally, the obtained mappings are transformed into the mappings among fuzzy ontologies. Experimental results with some fuzzy ontologies from the real world indicate that FOM-CG performs encouragingly well.

Key words: fuzzy ontology mapping; reference ontology; conceptual graph
Communicated by: G.-J. Houben & S. Auer

1 Introduction

The term ontology [19] is firstly used in the area of philosophy, and it means a "particular theory about the nature of being or the kinds of existents. Although first used in the area of philosophy, the term ontology has been used by researchers in a variety of areas such as artificial intelligence (AI),

information retrieval (IR), database theory, and the Semantic Web [2]. In the Semantic Web, ontologies are generally recognized as an essential knowledge base by providing a common understanding of a domain of interest. It allows communication and knowledge sharing among distributed users and applications. Meanwhile, the number of ontologies increases at a very rapid rate. But many ontologies in the same specific application domain may be similar to each other, as they are always used to represent the same thing with different viewpoints or representation methods. To this end, ontology researchers define a new kind of ontology (i.e., reference ontology [11]) to provide common and shared knowledge for a given application domain. More specifically, reference ontology that is considered as a standard knowledge base is in advance created according to the knowledge on the application domain, and the other ontologies, which are called as source ontologies, must be defined by referring to the reference ontology. If no Reference Ontology is predefined in the domain, we create or find out a Reference Ontology that contains the most common and shared information about the domain.

Note that the conceptual formalism supported by the ontology structure is not sufficient for handling imprecise and vague information that is commonly found in many application domains. In order to solve this problem, fuzzy set theory proposed by L. A. Zadeh [24] is introduced into ontology to generate a new knowledge base, i.e., fuzzy ontology (FO) [5]. Due to the higher level of expression of fuzzy ontology, many researches have focused on it. To effectively manage and organize multiple fuzzy ontologies in an application domain, mappings (i.e. semantic relationships) among these fuzzy ontologies should be created. Nevertheless, many existing ontology mapping approaches (details refer to Section 2) are designed for two classic ontologies, which are not sufficient for multiple fuzzy ontologies. That is because that they can not deal with fuzzy information. In addition, they only consider relationships between two ontologies. For this purpose, this paper provides a new method for Fuzzy Ontology Mapping based on Conceptual Graph model [16], and we name it as FOM-CG. This method is more suitable for multiple fuzzy ontologies composed of reference ontology and source ontologies in an application domain. For the sake of convenience in creating mappings, FOM-CG firstly transforms reference ontology and source ontologies into sets of conceptual graphs: R-set and S-sets, respectively. Then, FOM-CG compares the generated conceptual graphs to create mappings that can be divided into two types: R-set to S-sets (R-S) mappings and S-sets to S-sets (S-S) mappings. Finally, according to the mappings among conceptual graph sets, FOM-CG obtains the mappings for multiple fuzzy ontologies. Note that, the premise of creating mapping for two conceptual graphs from different sets is that the similarity of them is greater than a predefined threshold provided by ontology expert. Therefore, FOM-CG also takes the similarity calculation method for conceptual graphs as the most important work during creating mappings.

Specifically, this paper makes the following contributions:

- FOM-CG is used to create mappings among multiple fuzzy ontologies in an application domain. To manage these fuzzy ontologies effectively, FOM-CG considers a fuzzy ontology that contains the most common and shared information as reference ontology and the other fuzzy ontologies as source ontologies.
- FOM-CG applies conceptual graph model to solve the problem of mappings among fuzzy ontologies. More specifically, it provides a set of rules to transform fuzzy ontologies (i.e., reference ontology and source ontologies) into conceptual graph sets (i.e., R-set and S-sets), and

then creates mappings among these sets by comparing the transformed conceptual graphs from different sets.

- FOM-CG divided mappings among conceptual graph sets into two kinds: R-S mappings and S-S mappings. All the S-sets are connected with R-set by R-S mappings, and these R-S mappings are used to create S-S mappings among S-sets.
- FOM-CG proposes a series of algorithms to compare two sets of conceptual graphs. However, these algorithms are based on the comparison of Relation-Entity pairs (RE-pairs) that are the basic elements of conceptual graph. The notion of RE-pair is firstly put forward by us, and the specific description of RE-pair is also provided in this paper.

The remainder of this paper is organized as follows: some related works about ontology and fuzzy ontology are given in Section 2. Section 3 provides background knowledge, such as the formal definitions of fuzzy ontology and conceptual graph. In Section 4, a new method for mappings among multiple fuzzy ontologies in an application domain is proposed, which is called FOM-CG (Fuzzy Ontology Mapping based on Conceptual Graph). An example of applying FOM-CG to solve a specific problem of fuzzy ontology mapping is given in Section 5. Finally, Section 6 shows the general conclusion.

2 Related work

At present, many methods about ontology mapping for classic ontologies have been done. For example, Doan [8] proposed an ontology mapping system GLUE that applies machine learning techniques to create semantic mappings for ontologies. Giunchiglia [10] presented an algorithm S-Match for ontology, which uses a semantic thesaurus WordNet [13] for semantic matching between textual descriptions of ontology entities (e.g. concepts and relations). Tous [21] applied a vector space model for representing RDF labeled directed graphs translated from ontologies and this model was applied to ontology mapping process. Eidoon [9] represented VBOM that modeled ontologies in a vector space, and estimated their similarity degree by matching their concept vectors. Jean-Mary [12] provided an algorithm of ontology mapping ASMOV, which uses four kinds of ontology information, such as lexical information, external information, internal information, and individual information. Tang [20] proposed the ontology mapping model of RiMOM based on risk model, which creates mappings to reduce the risk of searching ontology. Pan [14] transformed ontologies into Bayesian Networks, and created mappings of BNs instead of the mappings of ontologies. Zheng [26] created mappings for ontologies based on the information of structure and instances. Buche [3] took the ontology mapping problem as a rule application problem in the Conceptual Graph model. Croitoru [7] proposed a (di)similarity measure based on the content and the structure of two graphs transformed from ontologies.

Nevertheless, the above methods can not deal with uncertain information, in other words, they are not fit for fuzzy ontologies. To this end, many efforts have focused on how to incorporate fuzzy set theory [24] into ontology and its representation language OWL. In [17] and [18], Straccia respectively proposed the fuzzy versions of ALC (D) and SHOIN (D), the corresponding OWL DL, the syntax and semantics is also presented. Zhai [25] proposed a series of fuzzy ontology models with the help of intuitionistic fuzzy set and fuzzy linguistic variable ontologies. Unfortunately, despite the popularity of

fuzzy description logic and the model of fuzzy ontology, relative little literatures about fuzzy ontology mapping have been carried out, such as [23], [22] and [1]. In [23], Xu defined the least upper bounds for fuzzy concepts, and applied the approximate concept mapping approach. Wang [22] applied "sup-min" a function for calculating the similarity between fuzzy sets, as a fuzzy concept can be represented as a fuzzy set. Afef [1] applied different relationships such as subsume, equivalent and overlap between fuzzy concepts, and reduced the problem of mapping fuzzy concepts to unsatisfiability checking. However, when these methods are used to create mappings among multiple fuzzy ontologies in an application domain, some problems have arisen, such as large computation quantity for comparing entities belonging to different fuzzy ontologies. To this end, FOM-CG proposed by this paper considers Reference Ontology (RO) as a standard ontology for an application domain to generate and manage other fuzzy ontologies called Source Ontologies. In FOM-CG, the RO of an application domain is given by predefining by experts or abstracting from the existed fuzzy ontologies, and the focus of this method is on how to use RO to move up the efficiency of mappings among fuzzy ontologies.

3 Preliminary

In this section, we recall some preliminaries on the definitions of fuzzy ontology and conceptual graph.

3.1 Fuzzy ontology

Fuzzy ontology is generated by incorporate fuzzy set theory into classic ontology. Therefore, it is a kind of knowledge base that contains and deals with fuzzy/ vague/ imprecise/ uncertain information. Some notions for fuzzy ontology have been proposed in the past researches, such as [6] and [4]. Based on the previous efforts, we introduce the formal definition of Fuzzy Ontology for FOM-CG as follows.

Definition 1 (Fuzzy Ontology). The formal definition of Fuzzy Ontology (FO) is a quintuple (**FO** = {**FC**, **P**, **FR**, **I**, **A**}) where:

- **FC** is the set of fuzzy concepts that are represented by properties and instances with memberships. Each fuzzy concept $C \in FC$ is a fuzzy set on the domain of instances $C: I \rightarrow [0, 1]$.
- **P** is the set of fuzzy concept properties that are considered as the attribute domains of fuzzy concept. Each property $P \in P$ is a basic unit for constructing fuzzy concept.
- **FR** is the set of fuzzy relations. Each $R \in FR$ is a binary fuzzy relation with a membership degree [0, 1] between a fuzzy concept and an entity such as a fuzzy concept, a property, and an instance.
- **I** is the set of instances that belong to fuzzy concepts with membership degrees [0, 1]. Each $I \in I$ is constituted by attribute values with respect to the properties (i.e., attribute domains) of fuzzy concept.
- **A** is the set of axioms expressed in a proper logical language, for example asserting class subsumption, equivalence, more generally to (fuzzily) constrain the possible values of concepts or instances.

In traditional software development the project moves more clearly from a requirements/specification phase, through successive designs that are evaluated and refined, until the system is built. In Web development, there is far less clarity in these phases, with significant overlap.

Designs are part of the build process, and lead through evaluation to a modification of specifications. Designs become successively deeper, moving from flat screens to functional prototypes, and there is an unclear distinction between the design process and the specification process, as in Figure 1.

3.2 Conceptual graph

A conceptual graph (CG) is a visual knowledge representation formalism based on the semantic networks of artificial intelligence and the existential graphs of Charles Sanders Peirce. A CG can be considered as a bipartite graph where two kinds of nodes are used to stand for concepts and conceptual relations, respectively. It is a kind of knowledge model to represent complex logic relations among concepts. For example, Figure 1 shows a CG for the sentence "John is going to Boston by bus", in which the rectangles stand for concepts, and the circles stand for conceptual relations. An arc pointing toward a circle marks the first argument of the relation, and an arc pointing away from a circle marks the last argument. The arrowhead can be omitted, if the central word of a CG is identified.

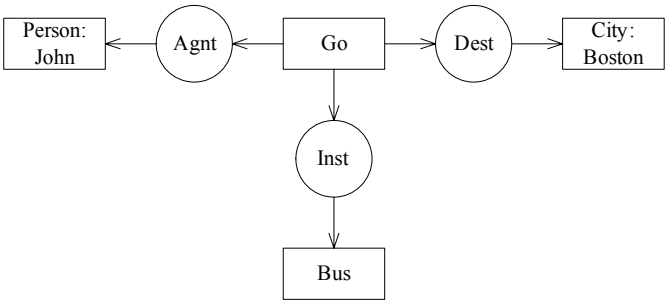

Figure 1 a CG for the sentence "John is going to Boston by bus"

Definition 3 (Conceptual Graph). The formal definition of Conceptual Graph (CG) is a 3-tuple **CG** = {**S**, **G**, **λ**} where:

- **S** = (T_C, T_R, I, *) is a support, where T_C is a finite, partially ordered set (poset) of concept types, TR a finite set of relation types, I a countable set of individual markers used to refer to specific concepts, * is the generic marker that refers to an unspecified concept of a specified type;
- **G** = (N_C, N_R, EG) is an ordered bipartite graph, where N_C is a set of concept nodes labeled by the concept types $\in$ Tc or the individual markers $\in$ I, N_R a set of conceptual relation nodes labeled by the relation types $\in$ T_R, and EG a set of edges;
- **λ** is a mapping function, which relats every node in the G to an element from the support S.

4 Applying Conceptual Graph to Fuzzy Ontology Mapping

In the section, we present a new method for Fuzzy Ontology Mapping based on Conceptual Graph, which is called FOM-CG. This method firstly translates multiple fuzzy ontologies (i.e., reference ontology and source ontologies) into sets of conceptual graphs (i.e. R-set and S-sets). In R-set and S-sets, a CG represents a fuzzy concept (FC) and all the other entities that are relative to the FC with fuzzy relationships. In this way, the problem of creating mappings among multiple fuzzy ontologies is

translated into the problem of how to create mappings among the CG sets. Next, two mapping sub-processes for FOM-CG are provided. In the first sub-process, mappings between R-set and S-sets are created, and we name these mappings as R-S mappings. In the second sub-process, R-S mappings are used to generate the mappings among S-sets, which are named by FOM-CG as S-S mappings. Finally, FOM-CG obtains mappings among multiple fuzzy ontologies according to the mappings among CGs. The main steps of FOM-CG are shown in Figure 2.

4.1. Generating conceptual graph sets: R-set and S-sets

The conceptual graph model has many advantages in application domains such as query and reasoning, especially in similarity calculation. In order to bring in the conceptual graph model to deal with the process of fuzzy ontology mapping, FOM-CG represents fuzzy ontologies by conceptual graph sets where each CG is represented by a FC and all the other entities connecting to the FC with relationships. However, the definitions for fuzzy ontology and the support of conceptual graph provided in Section 3 indicate that the two models have some similarities, although they represent knowledge by different ways. For this reason, we provide the transformation method from a fuzzy ontology to a support. Then, using the support and the structure of fuzzy ontology FOM-CG generates CGs. Therefore, the process of representing can be divided into two steps that are respectively used to generate the support and CGs based on structure of fuzzy ontology.

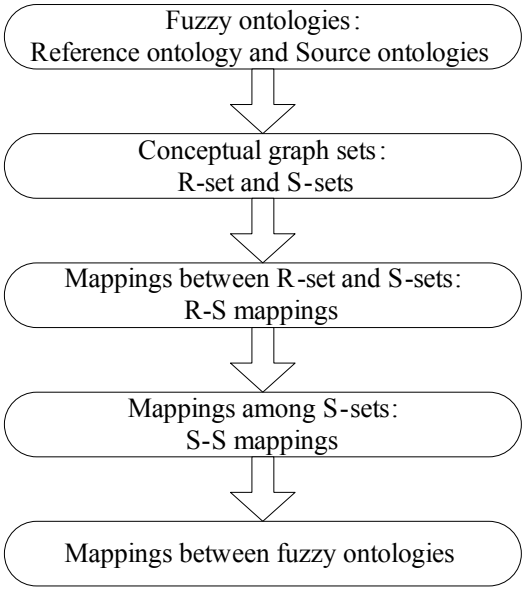

Figure 2 The process of FOM-CG

In the first step, a transformation function (φ) is usually used, and the specific transformation rules are as follows:

(1) $TC = \varphi(FC) \vee \varphi(P)$, i.e., fuzzy concepts (FC) and properties (P) in fuzzy ontology are all translated to concept types of the support, and organized by hyponymy to form a hierarchy (TC).

(2) TR = φ(FR) ∨ φ(A), i.e., fuzzy relationships (FR) and axioms (A) are all translated to relation types of the support. Note that, there are five kinds of relation types: "Is-parent-of", "Is-child-of", "Is-relative-to", "Has-instance", "Has-property".

(3) I = φ(I), i.e., instances of fuzzy concepts are translated to instances of concept types.

The above-mentioned transformation method is complete. To explain it, let us summarize the correspondence between the elements of fuzzy ontology and the elements of conceptual graph. It is easy to find that the five elements (i.e., FC, FR, P, I, and A) can be mapped into the corresponding elements in the support using the transformation function φ. More specifically, the mapping relationships are composed by FC ∨ P → TC, FR ∨ A → TR, and I → I. To sum up, the translation does not cause information loss.

In the second step, FOM-CG generates a conceptual graph set for a fuzzy ontology. Each conceptual graph in this set is composed of an entry, some entities, and relation types connecting the entry and elements. Now, let us give an example to explain how to represent a fuzzy concept and its adjacent entities (i.e., fuzzy concepts, instances and properties) as a CG.

Example 1: some information of fuzzy concept A in a fuzzy ontology is shown in Figure 3. Fuzzy concepts are represented by rectangles, the properties (P_1 to P_m) of A are represented by squares, and the instances (I_1 to I_n) belonging to FC A are represented by diamonds. Meanwhile, FC A is related to FC C_1 and C_2 with hypernym (Is-parent-of), related to FC P with hyponym (Is-child-of), and related to FC D with predicate-relationship (Is-relative-to). As related to the definition 1 for fuzzy ontology, all the relations are labeled by the membership α_i, where the subscript "i" represents the corresponding element.

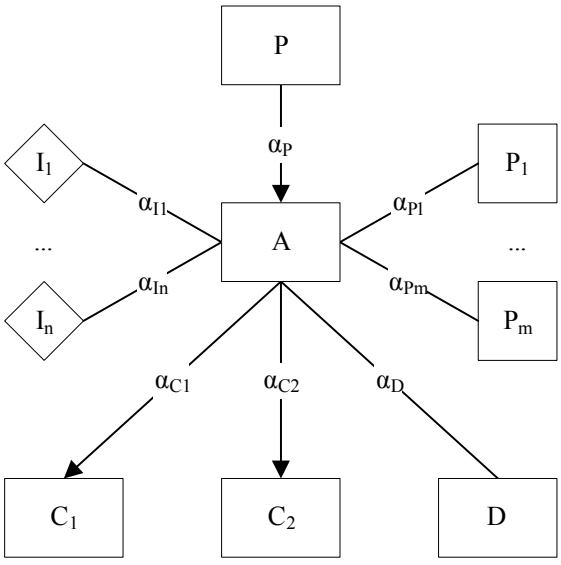

Figure 3 Fuzzy Concept A and its elements

It is easy to see that FC A is a central entity (i.e., entry) that is connected by all the other entities. Hence, FOM-CG takes FC A as an entry of CG, the objects relating to the entry as entities, and the connections between A and objects as relationships. Besides, instances are used to enrich the information of fuzzy concept, and they are connected by the relation type "Has-instance". The representation result is shown in Figure 4, where the rectangles are called concept nodes, and the ovals are called conceptual relation nodes. It is worthwhile to note that: entities in a CG are classified by the five kinds of relation types. This will bring a lot of convenience for FOM-CG to compare CGs and compute similarities for CGs in the next step.

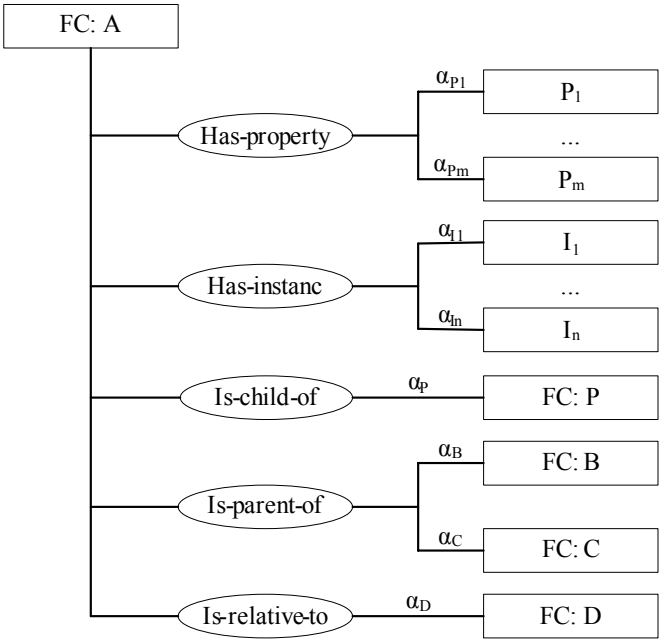

Figure 4 Conceptual Graph CGA translated from Fuzzy Concept A

Following the above method of generating a CG based on a fuzzy concept, a fuzzy ontology can be translated into a CG set. According to reference ontology and source ontologies as input, FOM-CG generates conceptual graph sets R-set and S-sets, respectively.

4.2. Creating R-S mappings between R-set and S-sets

There was a very strong feeling (average rating 4.4 on a 0-5 scale) that clients did not well understand the capabilities of the technologies. Similarly it was felt (average rating 4.2) that clients did not understand their own needs as they related to the technology. Perhaps surprisingly, anecdotal evidence indicated that respondents felt that clients had a low understanding of their own organisations and existing processes (most of the time undocumented) that need to be changed to allow for the effective integration of the new system. There was a majority consensus (i.e. 83% responding as *Strongly Agree* or *Agree*) that there needed to be a process at the beginning of the projects focussed on educating their clients.

As is known to all, many source ontologies are derived from the reference ontology by modifying, adding, or deleting some information. Obviously, the mappings between R-set and S-sets can be considered as clues that are useful for creating mappings among S-sets. In fact, an R-S mapping is a CG-pair that is made up of two CGs, where a CG is from R-set and the other one from S-sets. To the end, this section proposes a method to compare CGs and compute similarity for them.

In practice, the comparison of two CGs is difficult as many elements contained in the two CGs should be considered. To solve the problem, we divide each CG into fragments called Relation-Entity pairs (RE-pairs), and compare these RE-pairs to calculate the similarity for CGs. More specifically, a RE-pair is composed of a relation type and an entity, and the relation type is connected with the entity in CG.

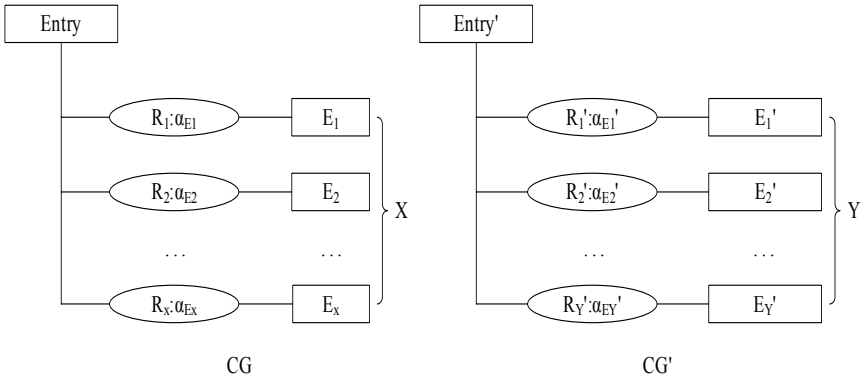

Figure 5 two CGs composed of RE-pairs

Example 2: Figure 5 shows two sketch maps of CG and CG', where CG contains x entities, and CG' contains y entities. After decomposing them we obtain many RE-pairs, and each pair is represented as a pair (Relation-type, Entity). To compare effectively, two RE-pair sets (i.e. $\{RE\}_X$ and $\{RE'\}_Y$) are created shown in Table 1, where X and Y are respectively the numbers of entities belonging to the two RE-pair sets. Thus, the problem of mapping between fuzzy concepts is translated to the problem of how to calculate the similarity for RE-pair sets.

Table 1 RE-pair sets

{RE}	{RE'}
(R_1: α_{E1}, E_1)	(R_1': α_{E1}', E_1')
(R_2: α_{E2}, E_2)	(R_2': α_{E2}', E_2')
…	…
(R_X: α_{EX}, E_X)	(R_X': α_{EX}', E_X')

Before explaining how to calculate the similarity of two sets of RE-pairs, we firstly provide the function "sim_p-p" to calculate the similarity of two RE-pairs. Take the data shown in Table 1 as an example, we can use the following formula to calculate the similarity between $RE_i = (R_i{:}a_{Ei}, E_i)$ belonging to $\{RE\}_X$ and $RE_j' = (R_j'{:}a_{Ej}', E_j')$ belonging to $\{RE'\}_Y$.

$$sim\_p-p(RE_i, RE_j') = \begin{cases} (\alpha_{Ei}/\alpha_{Ej}')*sim(R_i, R_j')*sim(E_i, E_j'), & \alpha_{Ei} < \alpha_{Ej}' \\ (\alpha_{Ej}'/\alpha_{Ei})*sim(R_i, R_j')*sim(E_i, E_j'), & \alpha_{Ej}' \le \alpha_{Ei} \end{cases} \quad (1)$$

Here, $sim(R_i, R_j')$ is the semantic similarities of relation types, and $sim(E_i, E_j')$ is the semantic similarities of entities. The calculation methods for $sim(R_i, R_j')$ and $sim(E_i, E_j')$ are respectively shown in formula (2) and (3). In the formula (2), the value of $sim(R_i, R_j')$ is 1 only if R_i is identical with R_j', otherwise, the value is 0. Previous transformation from fuzzy ontology to conceptual graph has provided the five kinds of relation types shown in Figure 4. Therefore, it is easy for us to compute the semantic similarities of relation types. However, the entities are varied, which contain more semantic information. Thus, formula (3) appears more complex, and the notion of information content (IC) [15] is introduced into it to calculate to the semantic similarities of entities. In the formula (3), E represents the least common ancestor of E_i and E_j' in the semantic lexicon such as WorldNet [13], and IC(E) = -log P(E), where P(E) is the probability of E.

$$sim(R_i, R_j') = \begin{cases} 1, & R_i = R_j' \\ 0, & R_i \neq R_j' \end{cases} \quad (2)$$

$$sim(E_i, E_j') = \frac{2*IC(E)}{IC(E_i) + IC(E_j')} \quad (3)$$

Next, we provide the definition of the "sim_p-s" algorithm for calculating the similarity of a RE-pair w.r.t. a set of RE-pairs. This algorithm compares the RE-pair (R_i:a_{Ei}, E_i) with all the RE-pairs in the set of {RE'}, and the objective of this algorithm is to find out the maximal value as the similarity of a RE-pair w.r.t. a set of RE-pairs.

Algorithm sim_p-s(RE-pair (R_i:a_{Ei}, E_i), RE-pair-set $\{RE'\}_Y$)

```
Similarity = 0
Foreach j = 1 to Y
        If Similarity < sim_p-p((Ri:aEi, Ei), (Rj':aEj', Ej'))
                Similarity = sim_p-p((Ri:aEi, Ei), (Rj':aEj', Ej'))
        EndIf
EndForeach
Return Similarity
```

Based on the above efforts, we propose the "sim_s-s" algorithm that is able to calculate the similarity of two sets of RE-pairs. This algorithm is implemented by calling the algorithm "sim_p-s". In details, a similarity for each RE-pairs in $\{RE\}_X$ w.r.t. $\{RE'\}_Y$ are calculated by "sim_p-s", and the similarity between $\{RE\}_X$ and $\{RE'\}_Y$ are calculated by the formula $(\sum_{i=1}^{X} similarity_i) / X$.

Algorithm sim_s-s(RE-pair-set {RE}, RE-pair-set {RE'})

```
Sum = 0
Foreach i = 1 to X
        Sum = Sum + sim_p-s((Ri:aEi, Ei), {RE'})
EndForeach
Similarity = Sum / X
Return Similarity
```

Now, the CGs in the R-set can be compared with the CGs in S-sets. The similarities between these candidate pairs, where a CG is from R-set and the other one is from S-sets, is computed by the algorithm "sim_s-s". Finally, according to the threshold of similarity given by experts, the mappings between R-set and S-sets are created. Moreover, the threshold is alterable to ensure that the accuracy and recall of FOM-CG could meet the requirement proposed by experts.

4.3. Creating mappings among CGs in S-sets

This section focuses on the similarity calculation for CGs in S-sets. However, this method is different from the method in Section 4.2. Concretely, the method can be divided into two parts. In the first part, all the CGs in S-sets are classified into groups according to the R-S mappings. The purpose of classifying is to put together the CGs that are similar to the same CG in R-set. The benefit of doing so is to avoid the comparisons for the CGs that have little common characteristics, and to move up the efficiency of comparison. During the grouping, FOM-CG labels all the RE-pairs of CGs in a group with the similar RE-pairs of CGs in R-set. In the second part, an algorithm "sim_c-c" is provided to compute the similarity of any two CGs belonging to the same group. The algorithm is implemented based on the comparisons of the labels of RE-pairs.

4.3.1 Classifying CGs in S-sets into groups by R-S mappings

For a CG (e.g. CG_A) in R-set, some CGs from different S-sets are mapped to it. To be convenience for explanation, we put these CGs mapping to CG_A into a group named as $Group_{CGA}$. According to the thinking of the "sim_s-s" algorithm, these CGs in the group are similar to CG_A in R-set. It is reasonable to suppose that these CGs may be similar to each other, as they possess some common characteristics hidden in CG_A. Conversely, given a CG (e.g. CG_B) in S-sets that is not mapped to CG_A, it is almost impossible that the CG_B is similar to any CG in the $Group_{CGA}$. To this end, in order to move up the efficiency of comparison for the CGs in S-sets, FOM-CG classifies the CGs in S-sets into groups according to the mappings from R-set to S-sets. More specifically, all the CGs in a group are mapped to the same CG of R-set, and the names of CGs in R-set are used to labeled these groups, respectively.

However, it is also a troublesome work to compare CGs in a group, because the group may contains many CGs, and any two CGs should be found out and compared. To solve this problem, FOM-CG labels all the RE-pairs of CGs in a group with pairs each of which contains an index and a similarity. The index can be used to find out the similar RE-pair of CGs in R-set. Thus, comparing RE-pairs can be realized by the comparison of their labels. More specifically, a label is composed of two parts, and represented as the pair (Index, Similarity). The following is an example to explain how to generate a label for a RE-pair.

Example 3: Suppose that CG_A that is a CG in R-set is similar to CG_B' that is a CG in S-sets, and a RE-pair (e.g. RE_i) belonging to CG_A is similar to a RE-pair (e.g. RE_j') belonging to CG_B'. The similarity of the two RE-pairs is represented by the capital letter "S" calculated by the algorithm "sim_p-p" in Section 4.2. When the CG_B' is put into the group $Group_{CGA}$, FOM-CG labels all the RE-pairs of CG_B' with the similar RE-pairs of CG_A and the similarity, such as RE_j': (A_i, S). Conversely, based on the "Index" (i.e. A_i) of label (A_i, S) of RE_j', we know that the i-th RE-pair in CG_A is similar to RE_j', and the similarity of the two RE-pairs is "S".

4.3.2 Comparing the labeled CGs in groups

After classifying, all the RE-pairs of CGs in a group are labeled by the same set of RE-pairs. It is easy to see that the labels of RE-pairs are not only simpler than RE-pairs but also easy to be compared. Therefore, the comparison of RE-pairs in groups can be replaced by the comparison of their labels. Next, FOM-CG illustrates how to compute the similarity of two CGs in a group by the comparison of labels of RE-pairs, in details.

Suppose that two CGs (CG and CG') are respectively composed of two RE-pair sets $\{RE\}_X$ and $\{RE'\}_Y$, where X and Y are the cardinalities of sets. The following algorithm "sim_c-c" can be used to compute the similarity of CG and CG'. In the algorithm "sim_c-c", some functions should be stated. The function Group(CG) returns the name of group that contains CG. The function Index(RE) returns a value that is the serial number of the similar RE-pair of CG in R-set. The function S(RE) returns the similarity of RE-pairs.

Algorithm sim_c-c(CG, CG')

```
Sum = 0
If Group(CG) == Group(CG')        //the compared CGs are from the same group
Foreach i = 1 to X
        Foreach j = 1 to Y
                If Index(REi) == Index(REj')
                        If S(REi) < S(REj')
                                Sum = Sum + S(REi) / S(REj')
                        Else
                                Sum = Sum + S(REj') / S(REi)
                        EndIf
                EndIf
        EndForeach
EndForeach
Similarity = Sum / Max(X,Y)
EndIf
Return Similarity
```

The algorithm "sim_c-c" has two characteristics: (1) only the CGs in the same group are allowed to be compared; (2) the similarity between two CGs is calculated by comparing the labels of their RE-pairs. Finally, according to the similarities calculated by Algorithm sim_c-c, FOM-CG creates mappings among S-sets (S-S mappings).

5 An example illustration

In this section, we apply the proposed method, i.e., FOM-CG, to create mappings among fuzzy ontologies that are on the domain of supermarket. Our goals are to introduce the mapping process of FOM-CG in details, to evaluate the matching accuracy of FOM-CG, and to verity that FOM-CG can be sufficient for creating mappings among multiple fuzzy ontologies in an application domain.

Figure 6 shows three fragments of fuzzy ontologies that are composed of a reference ontology and two source ontologies. Obviously, all the fuzzy concepts are connected by fuzzy relations (Is-child-of and Is-parent-of) with memberships, and properties and instances belong to fuzzy concepts by fuzzy relations (Has-property and Has-instance) with memberships. To be convenience for explanation,

FOM-CG uses letters in round brackets to label the corresponding fuzzy concepts, instances and properties.

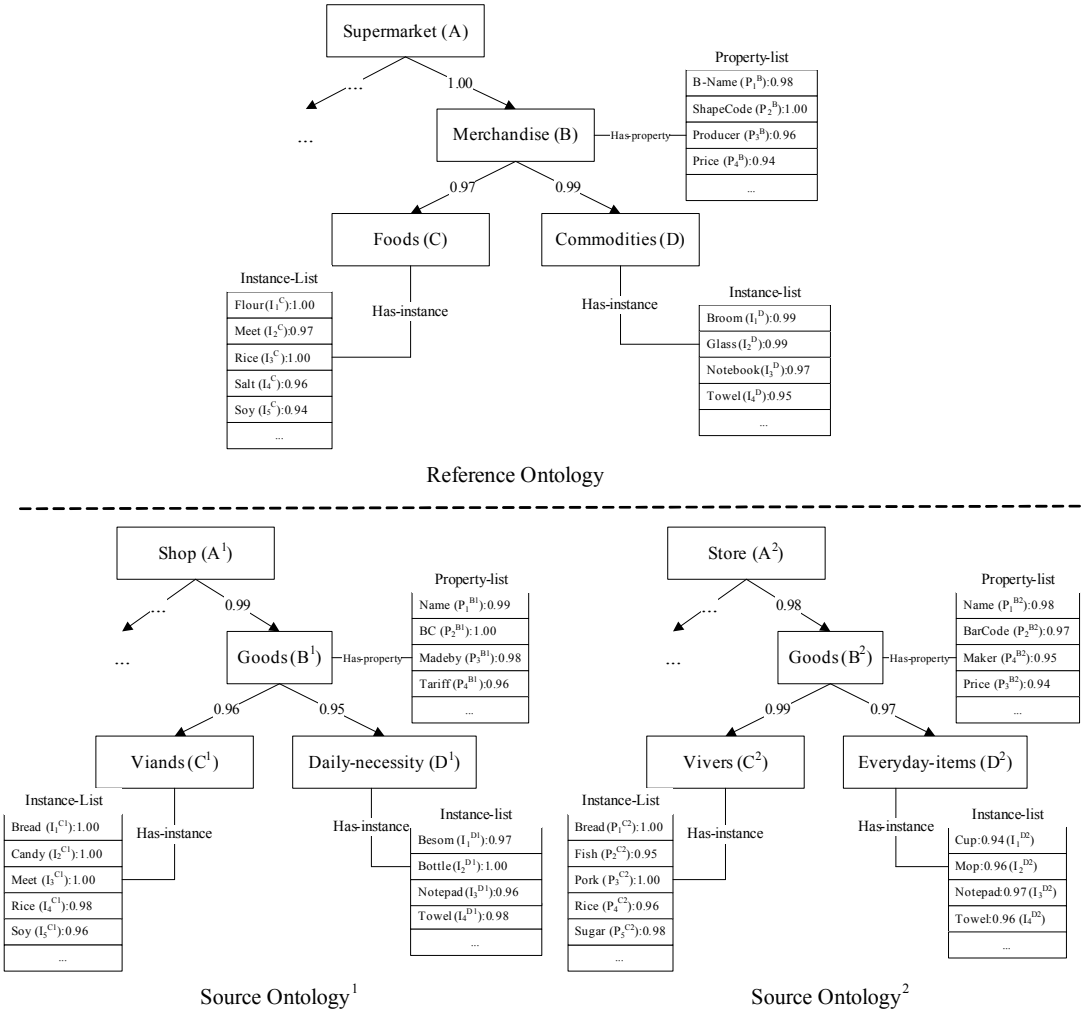

Figure 6 Multiple fuzzy ontologies containing Reference Ontology and two Source Ontologies

To creating mappings, FOM-CG firstly takes these fuzzy ontologies as inputs, and transforms them into conceptual graph sets: one R-set and two S-sets. The specific method for transformation is provided in Section 4.1, and the result is shown in Figure 7. In this figure, a conceptual graph is represented by a set of RE-pairs, and all the conceptual graphs belonging to the same CG set are stored in a table.

R-set	
Is-parent-of: 1.00, B	CG: A
...	
Is-child-of: 1.00, A	CG: B
Is-parent-of: 0.97, C	
Is-parent-of: 0.99, D	
Has-property: 0.98, P_1^B	
Has-property: 1.00, P_2^B	
...	
Is-child-of: 0.97, B	CG: C
Has-instance: 1.00, I_1^C	
Has-instance: 0.97, I_2^C	
...	
Is-child-of: 0.99, B	CG: D
Has-instance: 0.99, I_1^D	
Has-instance: 0.99, I_2^D	
...	

S-set^1	
Is-parent-of: 0.99, B^1	CG: A^1
...	
Is-child-of: 0.99, A^1	CG: B^1
Is-parent-of: 0.96, C^1	
Is-parent-of: 0.95, D^1	
Has-property: 0.99, P_1^{B1}	
Has-property: 1.00, P_2^{B1}	
...	
Is-child-of: 0.96, B^1	CG: C^1
Has-instance: 1.00, I_1^{C1}	
Has-instance: 1.00, I_2^{C1}	
...	
Is-child-of: 0.95, B^1	CG: D^1
Has-instance: 0.97, I_1^{D1}	
Has-instance: 1.00, I_2^{D1}	
...	

S-set^2	
Is-parent-of: 0.98, B^2	CG: A^2
...	
Is-child-of: 0.98, A^2	CG: B^2
Is-parent-of: 0.99, C^2	
Is-parent-of: 0.97, D^2	
Has-property: 0.98, P_1^{B2}	
Has-property: 0.97, P_2^{B2}	
...	
Is-child-of: 0.99, B^2	CG: C^2
Has-instance: 1.00, I_1^{C2}	
Has-instance: 0.95, I_2^{C2}	
...	
Is-child-of: 0.97, B^2	CG: D^2
Has-instance: 0.94, I_1^{D2}	
Has-instance: 0.96, I_2^{D2}	
...	

Figure 7 Conceptual Graph sets translated from fuzzy ontologies

Next, the algorithm sim_s-s is applied to calculate similarities for CGs in which a CG belongs to R-set and the other one belongs to S-sets. Take Conceptual Graph B and B1 as an example, FOM-CG compares all the RE-pairs of B with the RE-pair set of B1 in turn. In every comparison, the maximum similarity is added to the variable "Sum". Finally, the similarity between B and B^1 is calculated by Sum/X, where X is the number of RE-pairs of B. According to these similarities, FOM-CG creates R-S mappings shown in Figure 8, where mappings are labelled with the similarities.

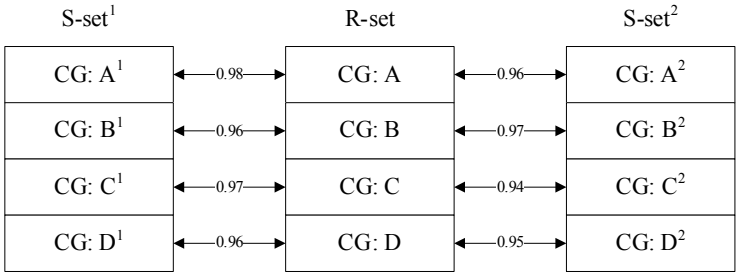

Figure 8 R-S mappings between R-set and S-sets

Next, FOM-CG puts CGs in S-sets into groups based on R-S mappings, and labels all RE-pairs of CGs in Sets. For example, CGs B^1 and B^2 in S-sets are mapped into B in R-set. Thus, they are put into $Group_B$, and their RE-pairs are labelled shown in Figure 9. Here, a label of RE-pair is composed of an index and a similarity. According to the index, FOM-CG can easily find out the similar RE-pair belonging to the CG in R-set. Meanwhile, the similarity can be used to create S-S mappings in the next step.

The labeled CG B^1

RE-pairs in CG B^1	Labels
Is-child-of: 0.99, A^1	(B_1, 0.97)
Is-parent-of: 0.96, C^1	(B_2, 0.94)
Is-parent-of: 0.95, D^1	(B_3, 0.96)
Has-property: 0.99, P_1^{B1}	(B_4, 0.94)
Has-property: 1.00, P_2^{B1}	(B_5, 0.96)
...	...

The labeled CG B^2

RE-pairs in CG B^2	Labels
Is-child-of: 0.98, A^2	(B_1, 0.95)
Is-parent-of: 0.99, C^2	(B_2, 0.96)
Is-parent-of: 0.97, D^2	(B_3, 0.98)
Has-property: 0.98, P_1^{B2}	(B_4, 0.97)
Has-property: 0.97, P_2^{B2}	(B_5, 0.99)
...	...

Figure 9 The labelled CGs B^1 and B^2 of S-sets in GroupB

Finally, FOM-CG applies the algorithm sim_c-c to compare the labelled CGs in the same group. In other words, any two CGs from different groups are not considered to be compared. Due to the more simple form of labels, the algorithm sim_c-c compares labels, not RE-pairs, to calculate similarities for the labelled CGs. Likewise S-S mappings are created by FOM-CG based on these similarities. Figure 10 shows the mappings among CG sets containing one R-set and Two S-sets, where solid lines represent R-S mappings and broken lines represent S-S mappings. Based on these mappings, FOM-CG generates mappings among the fuzzy ontologies.

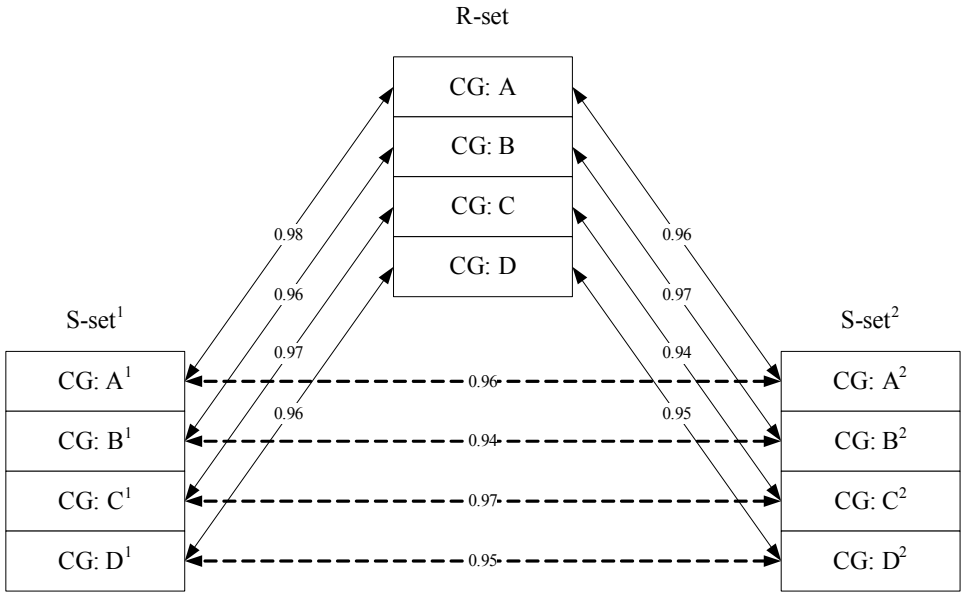

Figure 10 S-S mappings among sets of CGs

6 Conclusions

In this paper, we propose a new method for fuzzy ontology mapping based on conceptual graph (FOM-CG). It can be used to create mappings among multiple fuzzy ontologies in an application domain. This method firstly applies a conceptual graph model to represent a fuzzy concept and all the other

entities connecting to it with relationships. Thus, fuzzy ontologies, which contain reference ontology and source ontologies, are transformed into sets of conceptual graph (i.e. R-set and S-sets). The process of creating mappings in FOM-CG can be divided into two sub-processes, which are respectively creating R-S mappings between R-set and S-sets, and creating S-S mappings among S-sets. In the first sub-process, FOM-CG gives some algorithms to calculate similarity for conceptual graph sets, according to the characteristics of conceptual graph. In the second sub-process, FOM-CG uses R-S mappings to guide the mappings among S-sets. Our example demonstrates that FOM-CG is an effective method for fuzzy ontology mapping. Especially, it performs well to create mappings among multiple fuzzy ontologies in an application domain.

Note that our approach proposed in the paper is demonstrated on one real word problem and its usefulness is shown. In the near future we will implement our approach in prototype software and then test it with complex real-world application scenarios for its performance evaluation.

Acknowledgements

The authors wish to thank the anonymous referees for their valuable comments and suggestions, which improved the technical content and the presentation of the paper. This work was supported by the National Natural Science Foundation of China (60873010, 61073139 and 61202260), and in part by the Program for New Century Excellent Talents in University (NCET- 05-0288).

References

1. Bahri, A., Bouaziz, R. and Gargouri, F. Dealing with Similarity Relations in Fuzzy Ontologies. In: Proceedings of Fuzzy Systems Conference 2007, pp. 1-6, 2007.
2. Berners-Lee, T., Hendler, J. and Lassila, O. The Semantic Web. Scientific American 284(5), pp. 34-43, 2001.
3. Buche, P., Dibie-Barthelemy, J. and Ibanescu, L. Ontology mapping using fuzzy conceptual graphs and rules. In: Proceedings of International Conference on Computational Science 2008 (ICCS 2008). pp. 17-24, 2008.
4. Cai, Y. and Leung, H. F. A Formal Model of Fuzzy Ontology with Property Hierarchy and Object Membership. In: Proceedings of the 27th International Conference on Conceptual Modeling (ER 2008), Vol. 5231, pp. 69-82, 2008.
5. Calegari, S. and Ciucci, D. Towards a fuzzy ontology definition and a fuzzy extension of an ontology editor. In: Proceedings of the 10th International Conference on Enterprise Information Systems (ICEIS 2008), pp. 147-158, 2008.
6. Calegari, S. and Ciucci, D. Fuzzy ontology, fuzzy description logics and fuzzy-owl. In: Proceedings of International Workshop on Fuzzy Logic and Applications (WILF 2007), vol. 4578, pp. 118-126, 2007.
7. Croitoru, M., Hu, B., Dashmapatra, S., et al. A Conceptual Graph Based Approach to Ontology Similarity Measure. In: Proceedings of International Conference on Computational Science 2007 (ICCS 2007), pp. 154-164. , 2007.
8. Doan, A., Madhavan, J. and Domingos, P. Halevy A. Learning to Map between Ontologies on the Semantic Web. In: Proceedings of the Eleventh International World Wide Web Conference, pp. 662-673, 2002.
9. Eidoon, Z., Yazdani, N. and Oroumchian, F. Ontology Matching Using Vector Space. In : Proceedings of 30th European conference on Advances in information retrieval (ECIR 2008), pp. 472-481, 2008.

10. Giunchiglia, F., Shvaiko, P. and Yatskevich, M. S-Match: An algorithm and implementation of semantic matching. In: Proceedings of the European Semantic Web Symposium, pp. 61-75, 2004.
11. Goncalves, B., Guizzard, G. and Filho, J. Using an ECG reference ontology for semantic interoperability of ECG data. Journal of Biomedical Informatics, 43(1), pp. 126-136, 2011.
12. Jean-Mary, Y. and Kabuka, M. ASMOV Results for OAEI 2007. In: Proceedings of International Semantic Web Conference 2007 Ontology Matching Workshop, 2007.
13. Miller, G. A., Beckwith, R., Fellbaum, C., et al. Introduction to WordNet: An Online Lexical Database. International Journal of Lexicography, 3 (4), pp. 235-244, 1990.
14. Pan, R., Ding, Z., Yu, Y., et al. A Bayesian Network Approach to Ontology Mapping. In: Proceedings International Semantic Web Conference 2005, pp.563-577, 2005.
15. Resnik, P. Using information content to evaluate semantic similarity. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI95), pp. 448-453, 1995.
16. Sowa, J. F. Conceptual structures – Information processing in Mind and Machine. Addison-Welsey, 1984.
17. Straccia, U. Fuzzy description logics with concrete domains. In: Proceedings of the International Workshop on Description Logics (DL 2005), pp. 96–103, 2005.
18. Straccia, U. Towards a fuzzy description logic for the semantic Web. In proceedings of the 2nd European Semantic Web (ESWC 2005), pp. 167-181, 2005.
19. Studer, R., Benjamins, V. R. and Fensel, D. Knowledge Engineering: Principles and Methods. Data and Knowledge Engineering, 25 (122), pp. 161-197, 1998.
20. Tang, J., Li, J., Liang, B., et al. Using Bayesian decision for ontology mapping. Journal of Web Semantics 4 (4), pp.243-262, 2006.
21. Tous, R. and Delgado, J. A Vector Space Model for Semantic Similarity Calculation and OWL Ontology Alignment. In: Proceedings of the Seventeenth International Conference on Database and Expert Systems Applications (DEXA 2006), Vol. 4080, pp. 307-316, 2006.
22. Wang, Y., Zhang, R. B. and Lai, J. B. Measuring Concept Similarity between Fuzzy Ontologies. Fuzzy Information and Engineering, Vol. 2, pp. 163-171, 2009.
23. Xu, B. W., Kang, D. Z., Lu, J. J., et al. Mapping Fuzzy Concepts between Fuzzy Ontologies. In: Proceedings of 9th International Conference on Knowledge-Based Intelligent Information and Engineering Systems (KES 2005), pp. 199-205, 2005.
24. Zadeh, L. A. Fuzzy sets. Information and Control 8(3), pp. 338-353, 1965.
25. Zhai, J., Chen, Y., Wang, Q., et al. Fuzzy ontology models using intuitionistic fuzzy set for knowledge sharing on the semantic web. In: Proceedings of the 12th International Conference on Computer Supported Cooperative Work in Design (CSCWD 2008), pp.465–469, 2008.
26. Zheng, C., Shen, Y. P. and Mei, L. Ontology mapping based on structures and instances. In: Proceedings of International Conference on Machine Learning and Cybernetics 2010 (ICMLC 2010), pp. 460-465, 2010.