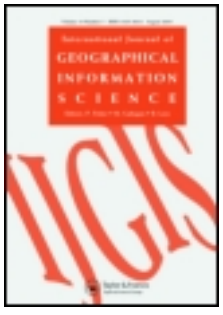

International Journal of Geographical Information Science

Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/tgis20

A node semantic similarity schema-matching method for multi-version Web Coverage Service retrieval

Nengcheng Chen [a], Jie He [b], Chao Yang [a] & Chao Wang [a]

[a] State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan, China

[b] School of Resources and Environment, Ningxia University, Ningxia, Yinchuan, China

Available online: 20 Mar 2012

To cite this article: Nengcheng Chen, Jie He, Chao Yang & Chao Wang (2012): A node semantic similarity schema-matching method for multi-version Web Coverage Service retrieval, International Journal of Geographical Information Science, DOI:10.1080/13658816.2011.647821

To link to this article: http://dx.doi.org/10.1080/13658816.2011.647821

International Journal of Geographical Information Science
i*First*, 2012, 1–22

Taylor & Francis
Taylor & Francis Group

A node semantic similarity schema-matching method for multi-version Web Coverage Service retrieval

Nengcheng Chen[a]*, Jie He[b], Chao Yang[a] and Chao Wang[a]

[a]*State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan, China;* [b]*School of Resources and Environment, Ningxia University, Ningxia, Yinchuan, China*

(*Received 8 July 2011; final version received 4 December 2011*)

Different versions of the Web Coverage Service (WCS) schemas of the Open Geospatial Consortium (OGC) reflect semantic conflict. When applying the extended FRAG-BASE schema-matching approach (a schema-matching method based on COMA++, including an improved schema decomposition algorithm and schema fragments identification algorithm, which enable COMA++-based support to OGC Web Service schema matching), the average recall of WCS schema matching is only 72%, average precision is only 82% and average overall is only 57%. To improve the quality of multi-version WCS retrieval, we propose a schema-matching method that measures node semantic similarity (NSS). The proposed method is based on WordNet, conjunctive normal form and a vector space model. A hybrid algorithm based on label meanings and annotations is designed to calculate the similarity between label concepts. We translate the semantic relationships between nodes into a propositional formula and verify the validity of this formula to confirm the semantic relationships. The algorithm first computes the label and node concepts and then calculates the conceptual relationship between the labels. Finally, the conceptual relationship between nodes is computed. We then use the NSS method in experiments on different versions of WCS. Results show that the average recall of WCS schema matching is greater than 83%; average precision reaches 92%; and average overall is 67%.

Keywords: schema matching; Web Coverage Service; semantic relationship; conjunctive normal form; vector space model

1. Introduction

Chen *et al.* (2011) proposed a fragment-based dynamic syntax schema-matching method (extended FRAG-BASE schema match) that is applicable to different versions of the Open Geospatial Consortium (OGC) Web Service schemas. When applying the extended FRAG-BASE schema-matching approach, the average *recall* of Web Coverage Service (WCS) (Whiteside and Evans 2008) schema matching is 72%; average *precision* is 82%; and average *overall* is 57%. Semantic heterogeneity of different OGC Web Service schemas manifests in two aspects: differences in element definitions (e.g. naming, additional, reduction and inheritance) and differences in attribute definitions (e.g. namespaces, constraint conditions and reference types).

*Corresponding author. Email: cnc@whu.edu.cn

ISSN 1365-8816 print/ISSN 1362-3087 online

http://dx.doi.org/10.1080/13658816.2011.647821
http://www.tandfonline.com

These differences considerably exacerbate the difficulty of schema matching. Such differences are observed in element naming (e.g. the same element label may have different meanings or different element labels may denote the same idea). In addition, the following factors also affect the difficulty of schema matching: for instance, the schemas have the characteristics of a complex element structure, are distributed and are of large scale. The proposed fragment-base schema-matching method in the syntactic level uses the division principle for a problem with a large schema to be matched, decomposes the schema files into small fragments and then matches the fragments. However, the syntactic schema-matching method is often unable to discover potential semantic mapping relationships, such as element 'abstract' and element 'description'; they have identical semantics, yet they cannot be identified by the syntax method. To improve schema-matching quality, particularly in terms of recall, we propose a new semantic schema-matching method based on the similarity between nodes. The method first computes the label and node concepts and then calculates the conceptual relationship between the labels to construct the propositional formula that represents the conceptual relationship between the nodes. Finally, we apply a propositional formula validator to verify the conceptual relationships. This study offers the following contributions:

(1) To overcome the compound words recognition, the proposed method calculated the semantic similarity between the different labels and assigned a weight to different semantic similarities with the label's importance in the node context.
(2) To improve the precision of schema matching, the node's annotation was used to calculate the similar value using the application of vector space model (VSM). The node semantic similarity (NSS) and node's annotation similarity were combined to generate the final similar value of correspondence schema. Compared to syntax schema-matching method, the proposed method achieved a higher matching accuracy.
(3) To improve the efficiency of schema matching, a schema division procedure was applied before semantic-matching execution; the proposed method decomposed the node labels into independent fragments. Compared to the general semantic schema-matching method, the proposed method achieved a higher performance.

The rest of the article is organised as follows: Section 2 provides an overview of semantic schema matching. Section 3 presents a comprehensive discussion of the NSS schema-matching method for multi-version WCS retrieval. Section 4 presents experiments on the quality of the COMA++, FRAG-BASE and NSS methods for WCS. Finally, Section 5 provides the conclusion and recommendations for future work.

2. Overview of semantic schema match

Semantic matching (Giunchiglia and Shvaiko 2003) computes semantic relationships (e.g. inclusion and equality) between the label concepts of elements (the original meaning of element labels) to obtain a value between the [0, 1] interval. Many semantic-matching systems for schema are currently available; these employ different algorithms to achieve semantic matching between schema elements.

First, there are some semantic-matching systems that calculate the words' semantic similarity based on WordNet (Miller 1995). For instance, COMA++ (Do and Rahm 2006) can perform general semantic matching, in which the semantic relationship between

elements is computed according to their semantic associations in WordNet. Typically, users are required to incorporate new semantic relationship pairs of elements into the COMA++ knowledge library. Therefore, COMA++ cannot precisely calculate semantic similarity, thereby yielding poor matching quality. Giunchiglia and Yatskevich (2007) propose an element-level semantic-matching approach using WordNet. The matchers use WordNet as a source of background knowledge and obtain semantic conceptual relationships from the database. The experimental results derived by the element-level matchers reflect as good a matching quality as that achieved by the matching systems of the Ontology Alignment Evaluation Initiative (OAEI 2006).

Second, there are many typical semantic-matching systems based on SATisfiability (SAT) solver, such as S-Match (Giunchiglia *et al*. 2004) and PROMPT (Noy and Musen 2000). S-Match (Giunchiglia and Shvaiko 2003) is an iterative semantic-matching system based on an SAT solver. It is an element-level semantic matcher that has a pair of schema files as input and pairs of matching elements as output. S-Match can improve matching precision and recall; however, schema attribute and instance have not been considered in this method. Its deficiency lies in its dependence on the efficiency of implementation, capacity for property processing and precision of the contextual meaning of a word.

Third, there are some semantic-matching experiments based on VSM (Algergawy *et al*. 2010). For instance, Carmel *et al*. (2002) applied VSM to query xml documents via xml fragments; the proposed method presented an extension of the VSM to integrate a measure of similarity between xml paths and embodied the model in information retrieval system. Rubén and Jaime (2006) used VSM to calculate semantic similarity for ontology alignment, modelled the ontology relationships with N-dimension vector space and applied a graph matching algorithm to compute the similarities between ontologies.

On the basis of analysis of the existing semantic-matching approaches, we propose a semantic-matching approach based on node similarity. The proposed approach is also designed according to the characteristics of the matching schema files. The algorithm enables the semantic analysis of label concepts, especially combination labels. In view of the varying glyphs and meanings of the different versions of schema label elements, we present a hybrid approach based on label meanings and annotations to calculate the similarity between label concepts. That is, we consider the use of generic terms (based on the semantic similarity approach) to calculate the similarities in label meanings and employ an approach based on a VSM to calculate the similarities between annotation labels. Results show that the approach is valid for labels with differences in glyph and meaning but similarities in annotation. The validity of the approach for such types of labels enables the discovery of new similarity relationships and effective improvement of recall. The semantic relationships between nodes are translated into a propositional formula, whose validity is then verified to confirm the semantic relationships between nodes.

3. Methods

Semantic schema matching is based on two key notions: ‘concept at a label’ and ‘concept at a node’. Label refers to a brief description of the element in the schema; it is an identifying or descriptive marker that is attached to an object. Node refers to a connecting point at which several lines come together; it is the source of lymph and lymphocytes. Concept at a label represents a set of documents or data instances belonging to the label. Concept at a node represents a set of documents or data instances that we would classify under this node.

3.1. Architecture

Figure 1 shows the overview of the semantic similarity schema-matching approach. The approach comprises two core and three complementary components. The two core components are schema preprocessing and matching implementation. The three complementary components are knowledge and data management libraries, a semantic matcher and an SAT solver.

The schema preprocessing component is used to analyse, represent and identify the input-distributed schema files into a series of independent fragments and also calculate the concepts of labels and nodes. The tasks in the component include (1) analyse and represent the input schema file using a common tree; (2) divide the common tree into a series of independent fragments (Chen *et al.* 2011); (3) calculate the concepts of all the labels in the schema tree according to the auxiliary information in the knowledge and data management libraries; and (4) calculate the concept of nodes in the schema tree.

The matching actuator computes the conceptual semantic relationships between all the label elements and nodes in the schema tree. On the basis of the labels in WordNet and annotation text in the VSM, we apply a hybrid method to calculate the conceptual relationships between labels. This method features combined meaning and annotation text similarities between labels. First, the conceptual relationships between nodes are calculated according to the conceptual relationship between labels. Second, the corresponding relationships between schema elements are determined according to the semantic relationships between nodes. The matcher used in the semantic-matching system includes string, meaning-based and annotation-based matchers. The meaning-based and annotation-based matchers input two WordNet meanings, after which the semantic relationships between meanings are calculated using WordNet hierarchy attributes and the annotation comparison technique. In calculating the semantic relationships between nodes, the computation of the conceptual relationships between nodes is translated into a kind of conjunctive normal form (CNF). The relationships between nodes are then verified using the propositional SAT solver. After this, the semantic similarity between nodes is calculated according to the concepts at nodes. Finally, the nodes that exceed the threshold are selected for outputting as mapping results. The semantic schema-matching approach based on node similarity inputs two matching schema files and outputs the mapping between the elements in the schema files.

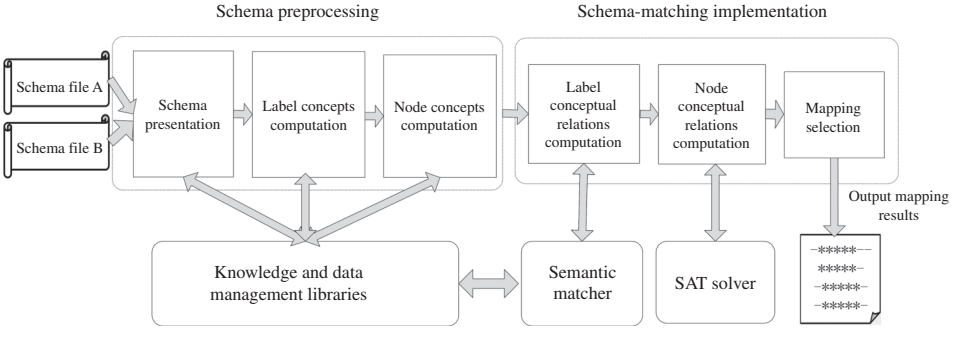

Figure 1. Overview of NSS schema-matching method.

3.2. Schema preprocessing

The main goal of schema preprocessing is to dynamically resolve the input schema and represent it using the schema tree. It also calculates the concepts at labels and nodes in the tree. If the internal schema is indicated as a graph structure, the Protoplasm tool is uniformly used to transform the schema into a tree structure (Bernstein *et al*. 2004). The OGC Web Service schema files include a distributed namespace and import other schema files through redirection (e.g. include and import). During the resolution process, the included or imported schema files are first downloaded to the local. Then, the corresponding referenced contents are imported into a separate file. Finally, the internal schema tree is used to represent the contents. After loading the represented schema, the concepts at labels and concepts at nodes of the schema elements are calculated through the following steps:

Step 1: Constructing the concepts at labels. The idea is to translate natural language expressions into internal formula language. The concepts and internal relationships are calculated according to the possible meanings of label words. For any given label, the concepts at labels, denoted as c_L, are returned through the analysis of semantics in the real world. We use propositional logic formula to encode the concept at labels. The labels are divided into words according to punctuation marks (e.g. space and character case); an example is the GetCoverage $\rightarrow$ <get, coverage> function. Usually, the computer can automatically handle this phase. Symbolised terms are mainly a complete set of characters or words, which can be found in WordNet. However, some terms cannot be found in the database; these include abbreviations (e.g. CRS and SRS). In such cases, the complete forms or meanings of terms should be determined according to annotations.

Second, lexical entries are classified and then analysed in terms of morphology to identify the various possible label forms; an example is Coverages $\rightarrow$ coverage. Lexical entry classification often requires manual inspection because a lexical entry may come in many forms (e.g. past tense or gerunds), thereby necessitating an analysis of the most primitive form of the word. To do this, we establish the atomic concept of a lexical entry and then use WordNet to extract the meaning of the classified lexical entries. For example, the label 'GetCoverage' with lexical entry 'get' has 36 verb meanings in WordNet, whereas 'coverage' has three noun meanings. Finally, a complex concept is constructed according to the preposition and conjunction in the label. For example, '$\cap$' represents conjunction 'or' and '$\cup$' denotes conjunction 'and'. We define the concept of a label thus:

Definition 1: The concept of a label. Suppose we have a label $A = (a_1, a_2, a_3, ..., a_i)$, where a_i is the lexical entry composition of label A, i is the lexical entry number of label A and the concept of label A is the combined meaning of all lexical entries in WordNet (i.e. a conjunction or disjunction of the meaning of all lexical entries). The formula is expressed as follows:

$$C_A = \left(a_1, WN_{a_1}\right) \propto \left(a_2, WN_{a_2}\right) \propto (a_3, WN_{a3}) \propto \cdots \propto \left(a_i, WN_{a_i}\right) \tag{1}$$

where WN_{a_i} represents the meaning with a combination of lexical entries of labela_i and $\propto$ denotes the combined operation symbol, which can be '$\cup$' or '$\cap$'. If the label A is decomposed by the word 'and' or character case, the combined operational symbol is '$\cup$'; else if the label A is decomposed by the word 'or' or space, the combined operation symbol '$\cap$' will be used.

Using GetCoverage label as an example, the concept is represented as follows:

$$C_{GetCoverage} = \left(get, WN_{get}\right) \cup \left(coverage, WN_{coverage}\right)$$

where '∪' is the disjunction of the meaning between *get* and *coverage* in WordNet. Another example is the label $C_{integerOrNull} = \left(integer, WN_{integer}\right) \cap (null, WN_{null})$
where '∩' is the conjunction of the meaning between *integer* and *null* in WordNet.

Step 2: Constructing the concepts at nodes. The concepts at nodes are based on the concepts at labels. In addition to considering the concepts at labels of the current node, the intermediate nodes from the root to the current node are identified. The idea is that by understanding the tree structure (i.e. the concept of a given label-arisen context), the concepts at labels can be expanded to concepts at nodes. C_n represents the concepts at nodes, where n indicates the position of the node in the tree. The concept of a node is defined as follows:

Definition 2: The concept of a node. Suppose we have a label with node A, and its position in the tree is n. $B_1, B_2, B_3, ..., B_n$ are intermediate nodes from the root to node A, and n is the number of intermediate nodes. Hence, the concept of node A is one of conjunctive concepts at labels of all the nodes from the root to node A. The formula is expressed as follows:

$$C_n = c_{B_1} \cap c_{B_2} \cap c_{B_3} \cap \cdots \cap c_{B_n} \cap c_A \tag{2}$$

where c_{B_n} represents the concepts at labels of intermediate nodes and c_A denotes the concepts at labels of node A.

Under normal conditions, if the tree where the node is located is unspecified, the default node refers to the node in the current tree. If there is more than one tree, subscript figures are typically used to indicate which node is located in which tree. For example, C_{2_5} represents the fifth concept at a node in the second tree. Using Figure 2 as an example, the seventh concept at a node in the first tree is expressed as follows: C_{1_7}=$C_{GetCoverage} \cap C_{rangeSubset} \cap C_{axisSubset} \cap C_{name}$.

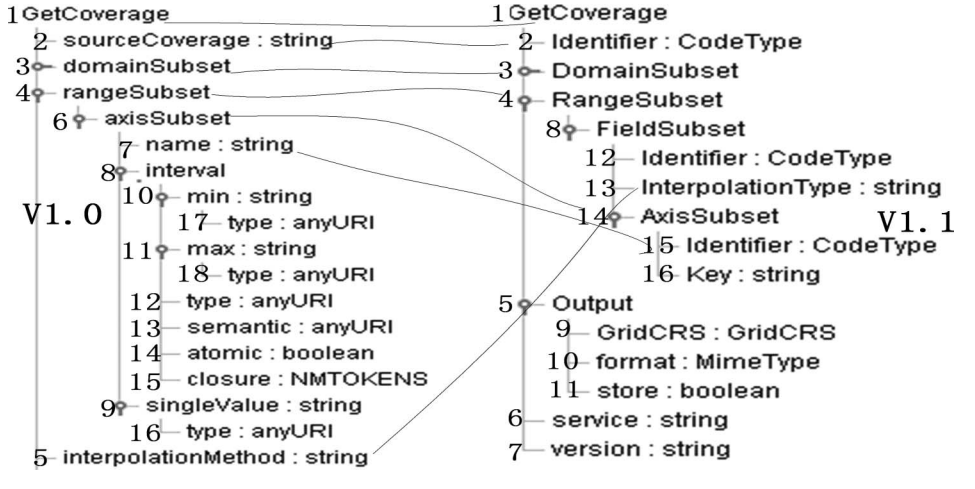

Figure 2. Schema mapping results in different versions of 'wcsGetCoverage.xsd' schema files.

3.3. Implementation of semantic schema matching

The core semantic matching uses the relationship of the concept at a label and the concept at a node to represent the relationship between two schema elements from the semantic of elements and schema structure. Then, it translates these relationships into similarity and finally generates optimal mapping according to the similarity. An element in one schema may correspond to more than one element in other schema. However, the matching cardinal is 1 to 1 according to the context of schema files. Therefore, we output the mapping with the highest similarity value. The procedure is called the optimal mapping. The conceptual relationships between nodes are *equivalence*, *more general*, *less general* and *disjointness*. When none of the relationships hold, the special *Idk* (I don't know) relationship is returned. *Equivalence* is the strongest relationship, with a similarity of 1. *Idk* is the weakest relationship, with a similarity of 0. *More* and *less general* are of the same strength, and their values can be selected by users. Each mapping element calculated by semantic matching is a four-tuple represented as follows:

$$< ID_{i,j}, n_{1i}, n_{2j}, R >, i = 1, 2, 3, ..., n_1, j = 1, 2, 3, ..., n_2$$

where $ID_{i,j}$ is the unique identifier of a mapping element, n_{1i} represents the ith node in the first schema tree, n_1 indicates the total number of nodes in the first schema tree, n_{2j} represents node j in the second schema tree, n_2 denotes the total number of nodes in the second schema tree and R is the concrete semantic relationship between two nodes. We take two versions of *WCS* schema files (Figure 2) as an example. The figure shows the schema diagram and part of the mapping results in two different versions of getCoverage.xsd schema files. Each node is identified with a unique number, and the letter 'C' is used to represent the concepts of nodes and labels. 'C_1' and 'C_2' denote the concepts of nodes and labels in the first and second trees. For example, $C_{1_{GetCoverage}}$ represents the concept at label *GetCoverage* and C_{1_3} represents the concept at label three in the first tree.

Semantic schema matching is divided into four steps, implemented in two phases. The first two steps are schema file parsing and the establishment of the concepts at labels and concepts at nodes. These steps are accomplished in the preprocessing phase. The third step is calculating semantic similarity between the concepts at labels. The fourth is computing semantic similarity between two nodes, which is completed in the semantic-matching implementation phase. The succeeding two sections focus on the computation of the similarity between the concepts at labels and similarity between the concepts at nodes.

3.3.1. Computation of the conceptual similarity between labels

Computing the conceptual similarity between labels requires the use of *a priori* knowledge (e.g. WordNet dictionary) and professional knowledge (which are domain oriented, for example, the term 'coverage' in the domain of earth science means geospatial data with spatial reference, such as satellite images, digital elevation model and other phenomena represented by values at each measurement point; but in news field it means newspapers, radio or television) to determine the semantic similarity between labels. The semantic organisational unit of WordNet is the tree, where the nodes are a synonym set with the same meaning and edges associated between synonym sets. The main types of association include synonyms and hyponyms. With the WordNet dictionary, we can decide the semantic relationship between labels quickly and improve the matching efficiency. On the basis of the label in the WordNet association between synonym sets, we can define the semantic relationship between labels.

Definition 3: Semantic relationship between labels. Suppose we have the following labels:

$$A = \left(A_{syns}, A_{hypers}, A_{hypos}, A_{meros}, A_{holos}, A_{antos}\right),$$
$$B = \left(B_{syns}, B_{hypers}, B_{hypos}, B_{meros}, B_{holos}, B_{antos}\right)$$

where A_{syns}, B_{syns}, A_{hypers}, B_{hypers}, A_{hypos}, B_{hypos}, A_{meros}, B_{meros}, A_{holos}, B_{holos}, A_{antos} and B_{antos} represent the synonym, hypernym, hyponym, metonym, homonym and antonym sets of labels A and B. Therefore, the semantic relationships between A and B are expressed as follows:

- $A = B$: When the hyponyms of a label in WordNet contains another label, the semantic relationship between these two labels is defined as *equivalence*, expressed by the formula $A_{syns} \supseteq B$ or $A \subseteq B_{syns}$, then $A = B$.
- $A \supseteq B$: On comparing labels in WordNet, whose hyponyms or metonyms contain another label, the semantic relationship between the two labels is defined as *more general*, expressed by the formula $A_{hopos} \supseteq B$ or $A_{meros} \supseteq B$, then $A \supseteq B$.
- $A \subseteq B$: On comparing a label in WordNet, whose hypernyms or homonyms contain another label, the semantic relationship between the two labels is defined as *less general*, expressed by the formula $A \subseteq B_{hopos}$ or $A \subseteq B_{meros}$, then $A \subseteq B$.
- $A \perp B$: On comparing a label in WordNet, whose synsets reflect any relationship or antonyms contain another label, the semantic relationship between the two labels is defined as *disjointness*, expressed by the formula $A_{antos} \supseteq B$ or $A \subseteq B_{antos}$, then $A \perp B$.

However, determining the exact meaning of a word in the current context is often difficult because one phrase may have many synonyms in WordNet. Therefore, estimating the semantic relationship according to the above-mentioned associative relationship (e.g. considering the semantic distance between synonyms and the amount of information included to determine semantic similarity) is insufficient for accurately determining the similarity between two labels. There are currently two kinds of WordNet-based similarity calculation methods. One is the hierarchical relationship-based semantic dictionary approach, in which word similarity is calculated according to the hyponymic and appositive relationship between these linguistic resources. Leacock *et al.* (1998) calculate the similarity between words using two synonyms in the WordNet hyponymic relationship that constitutes the shortest path. The dictionary-based approach is more intuitive, simple and effective. However, it is considerably affected by human subjectivity and sometimes cannot reflect objective fact. The other calculation method is based on the corpus statistics methods, which take the probability distribution of information context as a reference lexical similarity. Resnik (1999) uses information content to evaluate semantic similarity in taxonomy. Jiang and Conrath (1997) compute semantic similarity based on corpus statistics and lexical taxonomy. Lee (1993) computes word similarity using relative entropy, whereas Brown *et al.* (1991) use average mutual information. In addition, a context vector matcher (Patwardhan and Pedersen 2006) uses the semantic similarity between the context vectors of two synonyms to represent the similarity. A context vector matcher uses annotation vocabulary in WordNet as corpus. This task involves the following steps: (1) the frequency of words in the annotation vocabulary is calculated; (2) these words are searched in the related annotation in the corpus; (3) the dimension and amplitude of these word annotation vectors are computed; and (4) the vector dot product and magnitude of the two synonyms are used to represent the similarity. Quantitative analysis based on a statistical method enables the accurate and effective measurement of

the semantic similarity between words. However, this approach presents more complicated calculation and potentially problematic performance.

Problems are still encountered when calculating word similarity in WCS schema using WordNet. First, compound words and abbreviations are unsupported; therefore, WordNet cannot identify these words. For example, WordNet cannot identify *InterpolationMethod* from *Interpolation* and *Method* and *fun*, the abbreviation of *function*. Second, short terms are also unsupported. For example, WordNet cannot identify *CRS* from *Coordinate Reference System*. Third, professional and technical words are inadequately supported. For example, scalable vector graphics cannot be identified in WordNet. Fourth, the similarity between some labels cannot be evaluated based merely on the meaning of shapes. Examples are *sourceCoverage* and *identifier* or *BoundingBox* and *Envelope*. Although these labels are categorised as *disjointness* on the basis of the shape meanings, the label in the schema file annotation identifies them as associated with the semantic relationship.

In this study, most label elements of the matching schema files are compound labels, whose different versions may not have semantic association in meaning. Compound labels can be decomposed to compute combined semantic similarity. However, for compound labels with unrelated meanings but semantic association, semantic similarity cannot be accurately calculated. Moreover, the compound label elements in Web Service schema files described in the XML document schema have an annotation documentation (e.g. <xsl:annotation><xsl:document> annotation documentation </xsl:document></xsl:annotation>). These annotation documentations provide relatively accurate descriptions of the precise meaning of compound labels, serving as another reference tool for identifying the semantic relationship between two compound labels. For example, for labels *Identifier* and *sourceCoverage*, a low similarity equivalent to zero is obtained if only the shape meaning expressed in WordNet is considered. However, using the annotations of two labels as basis yields *Identifier* with the annotation '*identifier of the coverage that this GetCoverage operation request shall draw from*' and sourceCoverage with the annotation '*The coverage offering (Identified by its "name") that this request will draw from*'. Thus, the compound labels are similar in terms of annotation documentation and, accordingly, semantic relationship. To more accurately calculate the semantic similarity between labels, particularly that between compound labels, we adopt a similarity calculation approach. In this approach, the calculation of the similarity in meanings between compound labels and the annotation documentation of the labelling method between two labels are used as bases. That is, the meaning similarity and annotation documentation similarity are computed, and then the larger similarity is selected to represent the end similarity between two labels. This approach is based not only on the meaning of shapes, but also on the original meaning of the labels. This objectively reflects the definite semantic relationship between labels. Figure 3 shows the procedure for computing the similarity between labels. The steps are outlined as follows:

Step 1: Preprocessing label elements. The primary step is inputting two labels using a label processor after a collection of compositions is processed and an annotation of element labels is output in the schema files. The establishment of a collection of label compositions involves the processing of labels that symbolise and classify symbolic labels. After classification, each label composition can then be identified in WordNet.

Step 2: Computation of meaning similarity. Label compositions are entered using an approach based on semantic distance to calculate the meaning similarity between label compositions. Then, the meaning similarity between two labels is generated; that is, the similarity of concepts at labels is calculated.

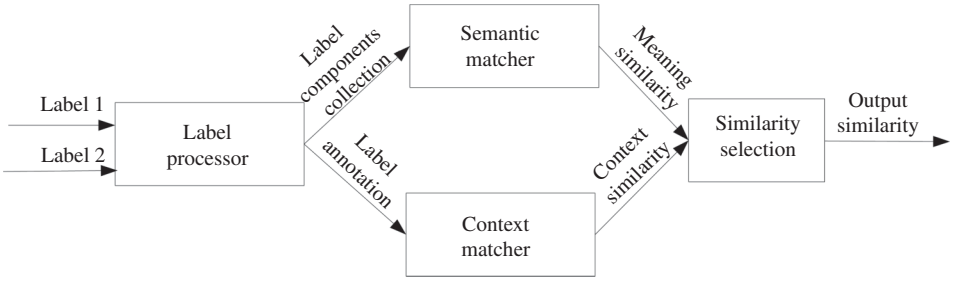

Figure 3. Procedure of similarity computing between labels.

Before calculating the similarity between label compositions, the atomic concepts of these compositions should first be established; that is, the meaning of each label composition should be searched in WordNet after classification. The meaning of each composition is called a sememe. Suppose the set of sememes is S. The number of sememes is represented as |S|, and each sememe is represented as S_i, where $i = 1, 2, 3,..., |S|$. The relationships of different compositions between sememes include hyponymy, synonymy and meronymy. All sememes constitute a sememe concept tree, where different levels of sememe meanings have different distances. For the distance between sememe S_i with hyponymy and its father sememe S_j, we apply the following formula:

$$\text{distance}\left(s_i, s_j\right) = \alpha + c_i \cdot \beta \tag{3}$$

where α represents the initial value of the distance, with the default value being 2.0, c_i denotes the depth of the sememe in the sememe concept tree and β indicates the decrement of each layer in the concept tree with a default value of –0.1. The distance between sememes with synonymy and meronymy is calculated according to the following formula:

$$\text{distance}\left(s_i, s_j\right) = w \cdot (\alpha + c_{\max} \cdot \beta) \tag{4}$$

where w represents the weight of a certain sememe relationship, and $w \geq 1$, $c_{\max}$ denotes the maximum depth of the sememe in the sememe concept tree. The depth is deeper in the sememe concept tree. The smaller the distance, the more similar the sememe. The Floyd algorithm is usually applied to calculate the shortest distance $dist(s_i, s_j)$ between S_i and S_j. Once the distance between two sememes has been calculated, we can compute the similarity between the two sememes. Rada *et al.* (1989) and Lee (1993) apply this approach to calculate similarity. The formula is expressed as

$$sim\left(s_i, s_j\right) = \frac{\lambda}{\lambda + dist\left(s_i, s_j\right)} \tag{5}$$

where λ is a non-vanishing constant $0 < \lambda < \infty$, often assigned a value of 1. $dist\left(s_i, s_j\right)$ represents the distance between S_i and S_j. For example, when the two sememes are the same, $dist\left(s_i, s_j\right)$ is 0, and the similarity between the sememes is $sim\left(s_i, s_j\right) = \frac{\lambda}{\lambda + dist\left(s_i, s_j\right)} = \frac{\lambda}{\lambda + 0} = 1$.

The calculation of the similarity between the compositions of the labels is based on the similarity calculation for the sememes of the compositions. Each composition is regarded as a word; the similarity between words is the similarity between the concepts of words. We then define the inner product of the concepts.

Definition 4: The inner product of the concepts. Suppose we have the following concepts:

$$C = (c_1, c_2, c_3, ..., c_m), D = (d_1, d_2, d_3, ..., d_n)$$

where $c_1, c_2, c_3, ..., c_m$ and $d_1, d_2, d_3, ..., d_n$ are the sememes of C and D, respectively. Then, the inner product of concepts C and $D$is represented as follows:

$$(C, D) = \sum_{i=1}^{m} \sum_{j=1}^{n} sim\left(c_i, d_j\right) \tag{6}$$

The normal number of the concepts and the definition of similarity between concepts can be derived according to the inner product of the concepts.

Definition 5: The normal number and similarity. Suppose we have a concept c, whose normal number is defined as

$$\|c\| = \sqrt{(c, c)} \tag{7}$$

The similarity between the two concepts is defined as follows:

$$sim(c, d) = \frac{(c, d)}{||c|| \cdot ||d||} = \frac{(c, d)}{\sqrt{(c, c) \cdot (d, d)}} \tag{8}$$

Equation (8) can be used to calculate similarity when dealing with the concept of a simple label. When compound labels are involved, the above-mentioned method is first used to calculate the similarities between the simple labels that constitute the compound labels. Then the similarities between the simple labels are combined to determine the similarity between two compound labels. Identifying key and non-critical parts is difficult. Some of these key and non-critical parts may be compound labels. Therefore, determining the importance of different sub-label weights is also a challenge. WordNet focuses only on describing nouns and verbs. Hence, we propose that the weights be determined according to the part of speech of the sub-label; that is, different weights are assigned to different labels with different parts of speech. The weights follow the order: $w_{noun} > w_{verb} > w_{adj} > w_{adv}$, where $w_{noun}, w_{verb}, w_{adj}$ and w_{adv} represent the weights of nouns, verbs, adverbs and adjectives. The calculation formula for the similarity between compound labels is defined as follows:

Definition 6: The similarity between labels. Suppose we have the following labels:

$$A = \left(a_{nouns}, a_{verbs}, a_{adjs}, a_{advs}\right),$$
$$B = \left(b_{nouns}, b_{verbs}, b_{adjs}, b_{advs}\right)$$

where a_{nouns}, b_{nouns}, a_{verbs}, b_{verbs}, a_{adjs}, b_{adjs}, a_{advs} and b_{advs} denote labels A and B after the set of nouns, verbs, adverbs and adjectives are symbolised. Thus, similarity $sim(A, B)$ between labels A and B is expressed as

$$\begin{aligned} sim(A, B) = {} & w_{noun} \times sim\left(a_{nouns}, b_{nouns}\right) + w_{verb} \times sim\left(a_{verbs}, b_{verbs}\right) \\ & + w_{adj} \times sim\left(a_{adjs}, b_{adjs}\right) + w_{adv} \times sim\left(a_{advs}, b_{advs}\right), \\ & w_{noun} + w_{verb} + w_{adj} + w_{adv} = 1.0 \end{aligned} \tag{9}$$

where $sim\,(a_{nouns}, b_{nouns})$, $sim\,(a_{verbs}, b_{verbs})$,$sim\left(a_{adjs}, b_{adjs}\right)$ and $sim\,(a_{advs}, b_{advs})$ calculate the similarity between the label pairs of nouns, verbs, adverbs and adjectives, respectively.

Figure 2 shows that we can calculate the similarity between compound labels *interpolationMethod* and *InterpolationType*. First, label *interpolationMethod* is symbolised into *interpolation* and *method*. *InterpolationType* is partitioned into *interpolation* and *type*. Then, the symbols are evaluated after being classified as parts of speech to obtain noun symbol pairs (*interpolation*, *interpolation*) and (*method*, *type*). Next, the similarity between noun symbol pairs is calculated using Equation (8). The similarity between (*interpolation*, *interpolation*) is 1.0 and that between (*method*, *type*) is 0.0. Finally, the similarity between compound labels *interpolationMethod* and *InterpolationType* is calculated using Equation (9):

$$\begin{aligned} sim(interpolationMethod, interpolationMethod) &= w_{noun1} \\ &\quad \times sim(interpolation, interpolation) \\ &\quad + w_{noun2} \times sim(Method, Type) \\ &= 0.6 \times 1.0 + 0.4 \times 0.0 = 0.6, \end{aligned}$$

where $w_{noun1} = 0.6$, $w_{noun2} = 0.4$.

Step 3: The computation of context similarity. We apply an approach based on a VSM to calculate the context similarity between the label annotations. To calculate the context similarity between the label annotations, we first create a VSM of annotation context. The VSM of the context is established as follows. First, word segmentation is applied to the context. That is, the words are segregated from the context according to the blank spaces between the English words in the context, and the high-frequency words in the context are removed according to *stopword* in WordNet. The root and affixes of each word are simultaneously removed. Context c_i is expressed as $\{w_1 w_2 w_3 ... w_k\}$, and the number of occurrences of each word N_{w_i} is calculated. Second, the weight vector of each word is calculated. Let us take words w_i as an example. We first query the Concept Chain of w_i in WordNet. Then, using the number of w_i occurrences, N_{w_i} is designated as a representation of the weight of its synonym and direct hypernym. Sequential processing of other words in the context is performed. If the WordNet synonym of some words w_j appears in the synonym set of w_i, change happens to the weight of synonym set corresponding to w_i; that is, N_{w_i}=N_{w_i}+N_{w_j}. When all the words in the context are disposed of, the weights of synonyms of all words are normalised. The normalised formula is shown as follows:

$$N_{w_i} = \frac{N_{w_i}}{\sqrt{\sum_{i=1}^{n} N_{w_i}^2}} \tag{10}$$

where N_{w_i} represents the weights of synonyms of some words and n denotes the number of words in the context.

When all the words in the context are disposed of, a context can be expressed using the VSM. For any context D, the final VSM is represented as follows:

$$\left\{w_{1u} n_{w_1}, w_{2u} n_{w_2}, \ldots, w_{mu} n_{w_m}\right\}$$

where w_{mu} is the context word vector and n_{w_m} is the word vector corresponding to the weight of the synonym.

An established document vector model constitutes a vector space M, where set $|M| = m$ for any word vector$w \in M$. The document space is expressed as $\{w_{1u}n_{w_1}, w_{2u}n_{w_2}, \ldots, w_{mu}n_{w_m}\}$. The inner product of the document space is defined below.

Definition 7: The inner product of the document space. Suppose we have the following documents:

$$d_i = \{w_{1i}n_{w_1}, w_{2i}n_{w_2}, \ldots, w_{mi}n_{w_m}\}, d_j = \{w_{1j}n_{w_1}, w_{2j}n_{w_2}, \ldots, w_{mj}n_{w_m}\}$$

then, the inner product of document d_i and d_j is expressed as

$$(d_i, d_j) = \sum_{u=1}^{m}\sum_{v=1}^{n}(w_{iu}n_{w_u}, w_{jv}n_{w_v}) = \sum_{u=1}^{m}\sum_{v=1}^{n} w_{iu}.w_{jv}.(n_{w_u}, n_{w_v}) \tag{11}$$

The document norm and the definition of context similarity can be derived according to the inner product of the document space.

Definition 8: Norm document and context similarity. Suppose we have the following document:

$$d_i = \{w_{1i}n_{w_1}, w_{2i}n_{w_2}, \ldots, w_{mi}n_{w_m}\}$$

The definition of its norm is represented as follows:

$$\|d_i\| = \sqrt{(d_i, d_i)} = \sqrt{\sum_{u=1}^{m}\sum_{v=1}^{m} w_{ui}.w_{vi}.(n_{w_u}, n_{w_v})} \tag{12}$$

Suppose we have the following document spaces:

$$d_i = \{w_{1u}n_{w_1}, w_{2u}n_{w_2}, \ldots, w_{mu}n_{w_m}\}, d_j = \{w_{1v}n_{w_1}, w_{2v}n_{w_2}, \ldots, w_{nv}n_{w_n}\}$$

then, the context similarity between document space $d_i$and d_j is defined as

$$sim(d_i, d_j) = \frac{(d_i, d_j)}{||d_i|| \cdot ||d_j||} = \frac{(d_i, d_j)}{\sqrt{(d_i, d_i) \cdot (d_j, d_j)}} \tag{13}$$

Let us take two labels, *Identifier* and *sourceCoverage*, as an example. Their annotations denote '*identifier of the coverage that this* GetCoverage *operation request shall draw from*' and '*The coverage offering (Identified by its "name") that this request will draw from*', respectively. The VSM is applied, first by performing word segmentation to obtain the word sets (*identifier*, *coverage*, *getcoverage*, *operation*, *request*, *draw*) and (*coverage*, *offering*, *identified*, *name*, *request*, *draw)*. Then, the word frequencies are calculated and expressed as (1,1,1,1,1,1) and (1,1,1,1,1,1), respectively. The word vector of these words are calculated and denoted as follows:

{*identifier*(0.0,0.0,0.56,0.12,0.0,0.0), *coverage*(1.0,0.0,0.0,0.11,0.0,0.0),
getcoverage(0.23,0.0,0.0,0.07,0.0,0.0), *operation*(0.0,0.0,0.0,0.0,0.0,0.0),
request(0.0,0.0,0.0,0.0,1.0,0.0), *draw*(0.0,0.0,0.0,0.0,0.0,1.0)},

{*coverage*(0.12,1.0,0.23,0.0,0.0,0.0), *offering*(0.0,0.0,0.0,0.0,0.0,0.0),
identified(0.56,0.07,0.0,0.00,0.0,0.0), *name*(0.12,0.11,0.0,0.0,0.0,0.0),
request(0.0,0.0,0.0,0.0,1.0,0.0), *draw* (0.0,0.0,0.0,0.0,0.0,1.0)}

Finally, the weights of these word vectors are calculated according to WordNet and represented as (0.17,0.18,0.17,0.16,0.16,0.16) and (0.18,0.16,0.16,0.18,0.16,0.16). Using Equation (13), we determine the annotation context similarity between labels as equivalent to 0.672.

Step 4: The selection of similarity. In calculating the meaning similarity between labels, as well as the annotation context similarity between labels, the maximum similarity is selected as the final similarity between labels. Hence,

$$sim(label1, label2) = \max\left(sim_{synonym}(label1, label2), sim_{gloss}(label1, label2)\right) \quad (14)$$

where $sim_{synonym}(label1, label2)$ represents the meaning similarity between two labels and $sim_{gloss}(label1, label2)$ expresses the annotation context similarity between two labels; if there are no annotations associated to the schema elements, then $sim_{gloss}(label1, label2) = 0.0$.

For example, the meaning similarity between labels *Identifier* and *sourceCoverage* is represented as $sim_{synonym}(identifier, sourceCoverage) = 0$, whereas the annotation context similarity is denoted as $sim_{gloss}(identifier, sourceCoverage) = 0.672$. Therefore, the final similarity between the two labels is 0.672.

Take Figure 2 as an example. Table 1 shows the final similarity, derived using Equation (14), and Table 2 shows the semantic relationship translated according to similarity.

To conveniently calculate the conceptual relationship between nodes as the similarity between all the labels is calculated, we translate the similarity between labels into a semantic relationship between labels. For example, if the similarity is equal to 1, then the similarity is defined as an *equivalence* relationship. If the similarity is less than 1 but higher than 0.6, then this is defined as a *more general* or *less general* relationship. A similarity of less than 0.5 is defined as a *disjointness* relationship. The relationship between labels is a matrix: one version of the node label is represented by the horizontal axis, and the other version is represented by the vertical axis, and the cell is used to represent semantic relationship value between two labels.

3.3.2. Computation of conceptual similarity between nodes

The similarity between nodes is equivalent to a semantic similarity between any two nodes in the schema tree, which is a quantitative description of semantic relationship between

Table 1. Similarity values between labels of 'wcsGetCoverage.xsd'.

v1.0 \ v1.1	GetCoverage	Identifier	DomainSubset	RangeSubset	InterpolationType	AxisSubset
GetCoverage	1.0	0.145	0.0	0.0	0.0	0.0
sourceCoverage	0.21	0.672	0.0	0.0	0.0	0.0
domainSubset	0.0	0.0	1.0	0.21	0.0	0.21
rangeSubset	0.0	0.0	0.21	1.0	0.0	0.21
axisSubset	0.0	0.0	0.21	0.21	0.0	1.0
Name	0.0	0.8	0.0	0.0	0.0	0.0
interpolationMethod	0.0	0.0	0.0	0.0	0.6	0.0

Table 2. Semantic relations between labels of 'wcsGetCoverage.xsd'.

v1.0 \ v1.1	GetCoverage	Identifier	DomainSubset	RangeSubset	InterpolationType	AxisSubset
GetCoverage	=					
sourceCoverage		⊆				
domainSubset			=	⊥		
rangeSubset			⊥	=		
axisSubset						=
Name		⊆				
interpolationMethod					=	

nodes. The idea in the computation of similarity between nodes is that if the relationship between nodes is represented using CNF, then the calculation of the relationship between nodes is translated into the issue of a verification normal formula. The computation of the similarity between nodes involves the following steps:

Step 1: The CNF of the conceptual relationship between nodes is constructed. That is, the conceptual relationship between two nodes is calculated according to the concepts of nodes. The concepts of nodes are established during the preprocessing of schema files, and the conceptual relationship formula for nodes is established on the basis of the conceptual axiom between nodes. First, we establish the conceptual axiom between nodes and define the axioms as follows:

Definition 9: Axioms. Suppose we have the following nodes: $A = \{a_1, a_2, \ldots, a_m\}$ and $B = \{b_1, b_2, \ldots, b_m\}$, and suppose $\alpha_1 = rel\,(a_1, b_1)$,$\alpha_2 = rel\,(a_2, b_2)$,...,$\alpha_m = rel\,(a_i, b_j)$ denote nodes A and B and the context semantic relationship between node labels, respectively. Then, the axiom between nodes A and B is

$$axiom(A, B) = \bigcap\nolimits_{i=1}^{m} (a_i \alpha_i b_i) \tag{15}$$

where a_1,a_2,...,a_m and b_1,b_2,...,b_n, respectively, represent the context concepts of node labels of nodes A and B.

Taking Figure 2 as an example, we calculate the axioms between nodes C_{1_5} and $C_{2_{13}}$. The node C_{1_5} context node labels are *GetCoverage*, *interpolationMethod*, and the node $C_{2_{13}}$ context node labels are *GetCoverage*, *RangeSubset*, *FieldSubset*, *InterpolationType*. The conceptual relationship between labels is *GetCoverage* = *GetCoverage*, *interpolationMethod* = *InterpolationType*. Therefore, *the axioms between nodes* C_{1_5} *and* $C_{2_{13}}$ *is (GetCoverage = GetCoverage)* ∧ *(interpolationMethod = InterpolationType)*. (∧ denotes the conjunction of conceptual relationships between nodes.)

Once the axiom is established, the conceptual relationship formula for nodes is set. However, before establishing the conceptual relationship formula for nodes, a kind of conceptual relationship between nodes should be set. Then, the validity of the relationships is verified; that is, we determine whether the relationships hold. The conceptual relationship between nodes is often assumed to contain the *more general* and *less general* relationships. When none of the relationships hold, the *disjointness* relationship is returned. When all of the relationships hold, the *equivalence* relationship is returned. Then, the data in Table 3 (second column) are used to translate the relationships of the concept at nodes into propositional conjunction.

The conceptual relationship between two nodes is derived through a series of axioms that work out the relationship establishment, using the formula

Table 3. Semantic relation and propositional formulas relation.

rel(*a*, *b*)	Translate *rel*(*a*, *b*) into propositional logic	Translate formula (8) into CNF
$a = b$	$A \leftrightarrow b$	N/A
$a \subseteq b$	$a \rightarrow b$	$axioms \wedge context1 \wedge \neg context2$
$a \supseteq b$	$b \rightarrow a$	$axioms \wedge context2 \wedge \neg context1$
$a \perp b$	$\neg (a \wedge b)$	$axioms \wedge context1 \wedge context2$

$$axioms \rightarrow rel\,(context1, context2) \tag{16}$$

where *axioms*, *context*1 and *context*2 are axioms, and the context of the first and second nodes, *rel*, is based on the axiom derived from the relationship between *context*1 and *context*2 (i.e. conceptual relationship between nodes). To confirm the validity of Equation (16), we show that its negation is not satisfied. That is,

$$axioms \rightarrow rel(context1, context2) \tag{17}$$

The third column in Table 2 describes how to use Equation (17) to translate the semantic relationship before testing.

We again take Figure 2 as an example. The relationship between C_{1_5} and $C_{2_{13}}$ should be verified. Suppose $C_{1_5} \subseteq C_{2_{13}}$, that is, $C_{1_5} \rightarrow C_{2_{13}}$. Then, we obtain the following relationship formula:

$$\begin{aligned}&(GetCoverage \leftrightarrow GetCoverage) \cap (interpolationMethod \leftrightarrow InterpolationType)\\ &\quad \cap (GetCoverage \cap interpolationMethod) \cap \neg (GetCoverage \cap RangeSubset\\ &\quad \cap FieldSubset \cap InterpolationType)\end{aligned} \tag{18}$$

We translate Equation (18) into CNF, expressed as follows:

$$\begin{aligned}&(\neg GetCoverage \cup GetCoverage) \cap (GetCoverage \cup \neg GetCoverage)\\ &\quad \cap (\neg interpolationMethod \cup InterpolationType)\\ &\quad \cap (interpolationMethod \cup \neg InterpolationType)\\ &\quad \cap (GetCoverage \cap interpolationMethod)\\ &\quad \cap \neg(GetCoverage \cap RangeSubset \cap FieldSubset \cap InterpolationType)\end{aligned} \tag{19}$$

A disjunction of all the atomic formula conjunctions is performed.

Step 2: Validating the propositional formula. All propositional formula validations use the SAT-based solver based on standard Davis-Putnam-Logemann-Loveland (DPLL). To confirm that Equation (19) holds, it is translated into

$$\begin{aligned}&(\neg GetCoverage \cup GetCoverage) \cap (GetCoverage \cup \neg GetCoverage)\\ &\quad \cap (\neg interpolationMethod \cup InterpolationType)\\ &\quad \cap (interpolationMethod \cup \neg InterpolationType)\\ &\quad \cap GetCoverage \cap interpolationMethod\\ &\quad \cap (\neg GetCoverage \cap \neg RangeSubset \cap \neg FieldSubset \cap \neg InterpolationType)\end{aligned} \tag{20}$$

Table 4. Conceptual relation between nodes of WCS.

	C_{2_2}	C_{2_3}	C_{2_4}	$C_{2_{13}}$	$C_{2_{14}}$
C_{1_2}	=	⊥	⊥	⊥	⊥
C_{1_3}	⊥	=	⊥	⊥	⊥
C_{1_4}	⊥	⊥	=	⊥	⊥
C_{1_5}	⊥	⊥	⊥	=	⊥
C_{1_6}	⊥	⊥	⊥	⊥	=

To simplify the calculation formula, all the bold parts in Equation (20) are assigned as true variables; that is, *context*1 in the formula assignment is true. The formula translated into Equation (20) is

$$GetCoverage \cap InterpolationType \cap (\neg GetCoverage \cap \neg RangeSubset \cap \neg FieldSubset \cap \neg InterpolationType) \tag{21}$$

No *context*1 variables are found in Equation (21). If the assignments of *GetCoverage, InterpolationMethod* are true, a contradiction exists in the results from the formula above. Therefore, Equation (18) is not satisfied and proves $C_{1_5} \rightarrow C_{2_{13}}$. Likewise, $C_{2_{13}} \rightarrow C_{1_5}$ can be confirmed. Finally, we derive $C_{1_5} \leftrightarrow C_{2_{13}}$, indicating that C_{1_5} and $C_{2_{13}}$ have an *equivalence* relationship.

Step 3: The conceptual relationship between nodes is translated into the similarity between nodes. In other words, different similarities are assigned to different types of conceptual relationship between nodes. For example, the assignment of the *equivalence* relationship is 1. The assignment of the *less general* or *more general* relationship is 0.8. Other relationships are assigned a value of 0. On the basis of the similarities, we generate the pair of nodes whose similarities are higher than the threshold as output mapping. Table 4 shows the results of parts of the conceptual relationship between nodes in Figure 2.

4. Results and discussion

There are seven schema files included in the WCS 1.0.0 (http://schemas.opengis.net/wcs/1.0.0/); we can find the three schema files 'wcsCapabilities.xsd', 'describeCoverage.xsd' and 'getCoverage.xsd' have covered the seven schema files. There are 17 schema files included in WCS 1.1.0 (http://schemas.opengis.net/wcs/1.1.0/); we also can find the 3 schema files 'wcsGetCapabilities.xsd', 'wcsDescribeCoverage.xsd' and 'wcsGetCoverage.xsd' have covered the 17 schema files. Therefore, the following schema-matching experiments are tested between 'wcsCapabilities.xsd' and 'wcsGetCapabilities.xsd', 'describeCoverage.xsd' and 'wcsDescribeCoverage.xsd' and 'getCoverage.xsd' and 'wcsGetCoverage.xsd'.

Three match methods, COMA++, FRAG-BASE and NSS, are applied in the following experiments. For COMA++, we use the best combination Strategies, that is, Average for aggregation, Both for direction, Average for computing combined similarity, and use the combination of Threshold and MaxDelta for selection. A domain dictionary containing 40 synonyms and 22 abbreviations is used in semantic name matching.

4.1. Results

Precision, recall and overall (Rahm and Bernstein 2001) measurements are employed to assess the matching quality of the schema-matching technique.

(1) *Precision* reflects the share of real correspondences between all the identified correspondences:

$$Precision = \frac{T}{P} = \frac{T}{T+F} \tag{22}$$

(2) *Recall* specifies the share of identified real correspondences:

$$Recall = \frac{T}{R} \tag{23}$$

(3) *Overall* represents a combined measure for match quality, taking into account the post-match effort needed to remove false matches and add missed matches:

$$Overall = Recall \times \left(2 - \frac{1}{Precision}\right) \tag{24}$$

where T represents the true matches, P denotes all the matches, F indicates the false matches and R specifies all the true matches, and T is a subset of R.

4.1.1. Precision

Figure 4 shows that the NSS method yields the best *precision* for wcsGetCapabilities, describeCoverage and getCoverage. The *precision* for wcsGetCapabilities is 95%; that for describeCoverage is 89%; and that for getCoverage is 91%. The mean precision of NSS is as high as 92%; that for COMA++ is about 83%; and that for FRAG-BASE is about 82%.

4.1.2. Recall

Figure 5 shows that the NSS method yields the best *recall* for wcsGetCapabilities, describeCoverage and getCoverage. The *recall* for wcsGetCapabilities is 78%; that for describeCoverage is 88%; and that for getCoverage is 84%. The mean *recall* of NSS is

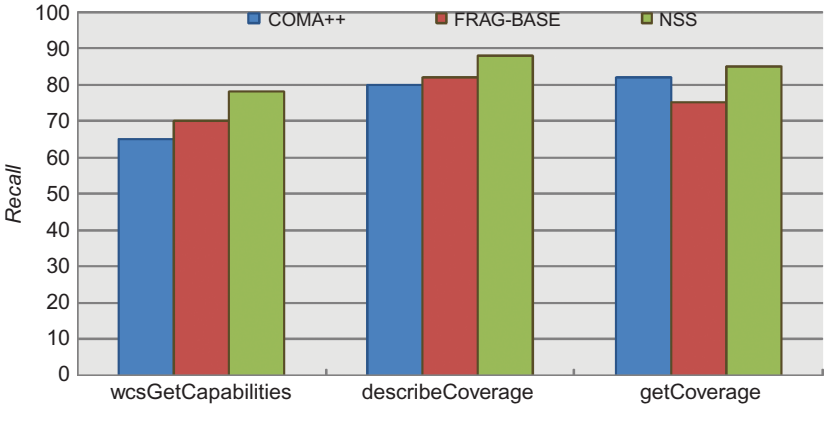

Figure 4. *Recall* of three match methods for WCS.

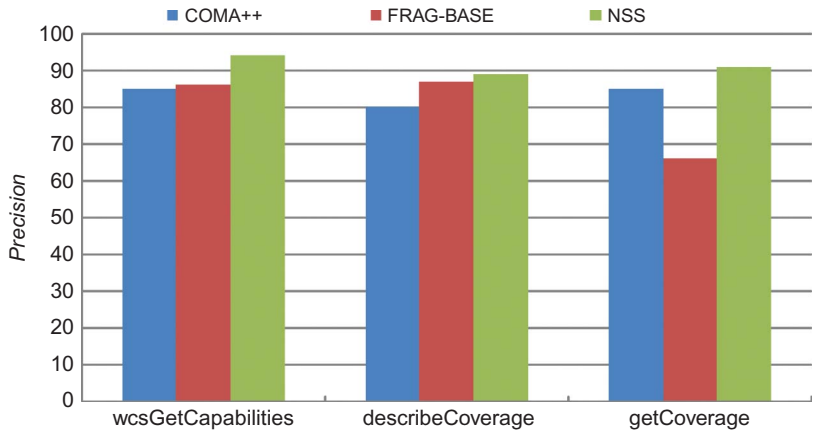

Figure 5. *Precision* of three match methods for WCS.

as high as 83%, whereas that for COMA++ and FRAG-BASE is about 75% and 72%, respectively.

4.1.3. Overall

Figure 6 shows that the NSS method yields the best *overall* for wcsGetCapabilities, describeCoverage and getCoverage. The *overall* for wcsGetCapabilities is 69%, whereas that for describeCoverage and getCoverage is 62% and 70%, respectively. The mean *overall* of NSS is as high as 67%, whereas that for COMA++ and FRAG-BASE is about 62% and 57%, respectively.

4.2. Discussion

4.2.1. Advantages

Compared with the conventional syntax-based matching method such as COMA++, the NSS method primarily performs semantic comparisons whilst offering syntax comparisons

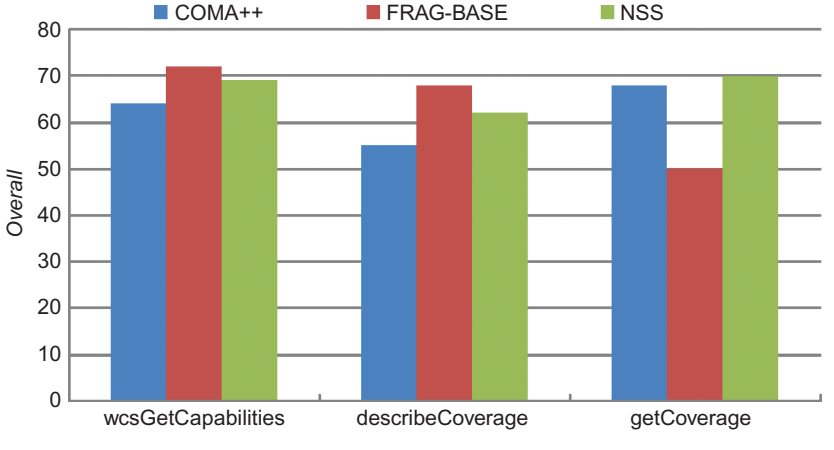

Figure 6. *Overall* of three match methods for WCS.

of labels that are unsupported by WordNet. Conventional syntax-based matching primarily performs syntax comparison, but does not carry out an in-depth analysis of semantic relationship between labels. Therefore, determining the mapping relationship between labels, which have semantic connection but different names, is difficult. In addition, expanding the concept of a label to that of a node, normalising the concepts of nodes and verifying the proposition of the normal formula result in the effective matching of elements from the schema element structure and context. A conventional syntax-based matcher such as CUPID performs structure matching mainly according to the tree where the nodes pass coverage for comparison without considering the context relationship between nodes. This is more purely a structural comparison and presents difficulty in reflecting the semantic relationship between nodes.

Compared with the conventional semantic-based matching method, the NSS method based on the similarity between nodes focuses on the semantic analysis of compound words and the weights of the composition of compound words. It more accurately calculates the semantic relationship between two compound words by applying a VSM to compute the annotation context similarity between label elements. This consequently improves the precision of the similarity calculation for two label elements and the recall of matching results.

4.2.2. Disadvantages

The experimental results show that when the semantic-based schema-matching method is used to match different schema files, the results vary considerably. These variations are attributed mainly to the difference in types, complexities and extent of difference between different versions. First, the more complex the structure of the schema elements and combination of label elements were, the longer the time it took to accomplish schema matching and the greater the probability of error matching and lost matching.

Second, different versions of schema files differ in terms of structural differences or semantics between the label elements because of the large differences between versions. The result is increased difficulty in achieving good matching quality. Compared with two different versions of WFS and WCS, WCS differs more considerably with respect to the class structure, property differences and elements named. These differences are mainly due to different versions of WCS, particularly the difference between the reference information models. These differences result in lower quality matching, when the semantic-based matching method is not applied. The schema matching of the two WCS versions yield an average recall of only about 70%, whereas the semantic-based matching method yields an average recall rate of 80%.

No in-depth analysis of performance issues is performed in the semantic-based matching method. The semantic-based schema-matching method is performed because of the need for additional preprocessing of all label elements (especially in processing the compound words), WordNet knowledge-based initialisation, semantic analysis, relationship identification and validation, calculation of similarity values and so on. Therefore, the efficiency is visibly lower than that achieved with COMA++ and other methods. More efforts are required for improvement.

5. Conclusion and future work

An NSS schema-matching method based on WordNet, CNF and VSM has been proposed for WCS. Given that different versions of the WCS schemas suffer from semantic heterogeneity, a hybrid algorithm was designed; this algorithm is based on label meanings

and annotations, enabling the calculation of the similarity between concepts of labels. Semantic relationships between nodes were translated into a propositional formula, and the validity of the formula was verified to confirm the semantic relationships between nodes. The NSS method was tested on different WCS versions. The results show that the average recall of the NSS matching is above 83%; average precision reaches 92%; and average overall is 67%. The proposed method overcomes the recognition of compound words and achieved a higher matching accuracy and a higher performance. It will effectively promote the integration of heterogeneous web services. Future research will focus on the time complexity analysis and run-time evaluation of the NSS method and developing a distributed NSS approach to improve the efficiency of the schema matching.

Acknowledgements

This work was supported in part by the National Basic Research Program of China (973 Program) under Grant 2011CB707101, by the 863 Program under Grant 2011AA010500 and by the National Natural Science Foundation of China under Grants 41171315, 41023001 and 41021061. The authors thank the anonymous reviewers for their valuable comments and insightful ideas.

References

Algergawy, A., Nayak, R., and Saake, G., 2010. Element similarity measures in XML schema matching. *Information Sciences*, 180 (24), 4975–4998.

Bernstein, P., *et al*., 2004. Industrial-strength schema matching. *ACM SIGMOD Record*, 33 (4), 38–43.

Brown, P., *et al*., 1991. Word sense disambiguation using statistical methods. *In: Proceedings of 29th meeting of the Association for Computational Linguistics (ACL-91)*, June 1991, Stroudsburg. Pennsylvania: ACL, 264–270.

Carmel, D., *et al*., 2002. An extension of the vector space model for querying XML documents via XML fragments. *SIGIR Forum*, 36 (2), 1–12.

Chen, N., *et al*., 2011. Extended FRAG-BASE schema matching for multi-version open GIS services retrieval. *International Journal of Geographical Information Science*, 25 (7), 1045–1068.

Do, H.H. and Rahm, E., 2006. Matching large schemas: approaches and evaluation. *Information Systems*, 32 (6), 857–885.

Giunchiglia, F. and Shvaiko, P., 2003. Semantic matching. *The Knowledge Engineering Review Journal*, 18 (3), 265–280.

Giunchiglia, F., Shvaiko, P., and Yatskevich, M., 2004. *S-match: an algorithm and an implementation of semantic matching*. Lecture Notes in Computer Science, 3053. Heidelberg: Springer, 61–75.

Giunchiglia, F. and Yatskevich, M., 2004. Element level semantic matching. In: *Proceedings of meaning coordination and negotiation workshop at ISWC*, 8 November, Hiroshima, Japan. Heidelberg: Springer, 37–48.

Jiang, J.J. and Conrath, D.W., 1997. Semantic similarity based on corpus statistics and lexical taxonomy. *In: Proceedings of the international conference on research in computational linguistics*, 1997, 19–33.

Leacock, C., Chodorow, M., and Miller, G.A., 1998. Using corpus statistics and WordNet relations for sense identification. *Computational Linguistics*, 24 (1), 147–165.

Lee, J.H., 1993. Information retrieval based on conceptual distance in ISA hierarchies. *Journal of Documentation*, 49 (2), 188–207.

Madhavan, J., Bernstein, P.A., and Rahm, E., 2001. Generic schema matching with Cupid. *In: Proceedings of the 27th international conference on very large data bases*, 11–14 September, Roma, Italy. San Francisco: Morgan Kaufmann, 49–58.

Miller, G., 1995. WordNet: a lexical database for English. *Communications of the ACM*, 38 (11), 39–41.

Noy, N.F. and Musen, M.A., 2000. PROMPT: algorithm and tool for automated ontology merging and alignment. *In: Proceedings of the seventeenth national conference on artificial intelligence (AAAI-2000)*, 3 August, Austin, TX: AAAI Press, 450, 455

Ontology Alignment Evaluation Initiative, 2006 [online]. Available from: http://oaei.ontologymatching.org/2006/ [Accessed 18 May 2011].

Patwardhan, S. and Pedersen, T., 2006. Using WordNet based context vectors to estimate the semantic relatedness of concepts. *In: Proceedings of the EACL 2006 workshop making sense of sense – bringing computational linguistics and psycholinguistics together*, 4 April, Trento, Italy. UK: Routledge, 1–8.

Rada, R., *et al*., 1989. Development and application of a metric on semantic nets. *IEEE Transactions on System, Man and Cybernetics*, 19 (1), 17–30.

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

Rubén, T. and Jaime, D., 2006. A vector space model for semantic similarity calculation and OWL ontology alignment. *LNCS*, 4080. Heidelberg: Springer, 307–316.

Whiteside, A. and Evans, J.D., eds., 2008. *OGC™ Web Coverage Service Implementation Standard (version 1.1.2)* [online]. Open Geospatial Consortium. Available from: http://www.opengeospatial.org/standards/wcs [Accessed 10 May 2011].