Pers Ubiquit Comput (2013) 17:1331–1339
DOI 10.1007/s00779-012-0567-5

ORIGINAL ARTICLE

Semantic and structural similarities between XML Schemas for integration of ubiquitous healthcare data

Pham Thu Thu Thuy · Young-Koo Lee · Sungyoung Lee

Received: 9 August 2011 / Accepted: 26 December 2011 / Published online: 10 July 2012

Abstract Currently, a lot of recent electronic health records are based on XML documents. In order to integrate these heterogeneous XML medical documents efficiently, studies on finding structure and semantic similarity between XML Schemas have been exploited. The main problem is how to harvest the most appropriate relatedness to combine two schemas as a global XML Schema for reusing and referring purposes. In this paper, we propose the novel resemblance measure that concurrently considers both structural and semantic information of two specific healthcare XML Schemas. Specifically, we introduce new metrics to compute the datatype and cardinality constraint similarities, which improve the quality of the semantic assessment. On the basis of the similarity between each element pair, we put forward an algorithm to calculate the similarity between XML Schema trees. Experimental results lead to the conclusion that our methodology provides better similarity values than the others with regard to the accuracy of semantic and structure similarities.

Keywords XML healthcare data · Semantics · Structure · Similarity · Measurement

P. T. T. Thuy · Y.-K. Lee (✉) · S. Lee
Department of Computer Engineering, Kyung Hee University, Seoul, South Korea
e-mail: yklee@khu.ac.kr

P. T. T. Thuy
e-mail: tttpham@oslab.khu.ac.kr

S. Lee
e-mail: sylee@oslab.khu.ac.kr

1 Introduction

Ubiquitous healthcare data are the collection of healthcare data from the large number of environmental and patient sensors and actuators to monitor and improve patient's physical and mental conditions [1]. Nowadays, the ubiquitous healthcare data are increasing, so the healthcare providers need to integrate these data in order to keep them as the electronic health record (EHR). EHR is a gathering of the electronic healthcare information about individual patients, and it is capable of being shared across different healthcare systems [2]. Therefore, the integration of healthcare data plays an important role in enhancing the quality of the patient care and the information exchange among the medical systems.

Healthcare data are often stored in XML format, since XML and its schema language have received a wide acceptance as a standard for representing and exchanging medical data among heterogeneous systems. With the data in XML format, healthcare service users can create medical documents easily and store them in the distributed computing environments. Developers can create their own XML documents, which obey some structure rules. These structure regulations are usually defined through XML Schema (XSD) and document type definition (DTD). XSD is used widely than DTD because of its advantages. XML Schema is itself an XML document. It is also more powerful than DTD in supporting the datatype and namespace definition [3]. For these reasons, XML Schema is more often used to define the structure for the healthcare data than DTD.

Healthcare data described using different XML Schemas, however, bring a challenging issue when integrating the data. In the real healthcare XML Schema documents, many elements in the schemas have an identical semantics,

but they may have the different name and structure. On the contrary, many elements have a similar model but are not similar in the meaning. Therefore, one of the main problems with integration of the healthcare data is how to assess the similarity of elements among XML Schema documents.

There are a lot of studies proposing the metrics for measuring the resemblance of concepts between two documents [4–11]. Nevertheless, most of them focused on measuring the name similarity or on the structural similarity of the elements between two documents. Some studies concentrated on both name and structure, but some factors in their metrics should be assigned manually using human's judgement [7, 11]. Moreover, the XML healthcare data are very large and contain varied definitions for datatypes and cardinality constraints, and they need formal metrics to obtain the accurate similarity values.

In this work, we propose the ESim measure, a fully automated process of measuring structural and semantic similarity between elements from both XSD healthcare trees. We evaluate our method with performance study and comparison to related works.

Our contributions are:

- We propose a new metric to measure the datatype similarity between two attribute types.
- We present the novel metric to measure the similarity of the cardinality constraints of the elements.
- In order to avoid the case that two nodes have the same structure but difference in their names, we compute the structural similarity of two concepts by relying on the semantic similarity and each pair of their direct children elements.
- We present an algorithm to calculate the similarity between two schema trees.
- We conduct a set of experiments to evaluate our computation and compare with other works.

The rest of the paper is organized as follows. Section 2 presents some specific methods of the related work. In the Sect. 3, we propose the similarity between XML Schema documents. An experimental study is described in Sect. 4. Finally, Sect. 5 summarizes the paper and mentions to the future research.

2 Related works

In this section, we present some approaches proposed in the related researches to measure the similarity between two simple concepts. One of the traditional methods in calculating this similarity is proposed by Rada [12]. They provided a formula to compute the resemblance between two concepts via the distance between them in a taxonomy tree.

Similarly, Fernandez et al. [13] and Li et al. [14] considered some degrees of similarity between concepts based on the path length between them. They assumed that the more numbers of nodes in two paths have the same name, the more similarity between two considered concepts.

Other proposals refine these above approaches by considering the depth of the concepts in the taxonomy. This makes sense with the assumption that concepts at upper layers have more general semantics and less similarity between them, while concepts at lower layers have more concreted semantics and thus stronger similarity. With this idea, Wu and Palmer [4] used the term "score" to describe the similarity between two elements.

In general, most of the previous works focus on concept similarity between two elements in the same taxonomy tree. And the distance between concepts is the main parameter used by a structural approach. However, distance method is commonly used when assuming that instances are equally distributed over concepts. These approaches are different from our aim and strategy, since we focus on measuring the similarity between elements in two XML Schemas. Moreover, the distance of concepts in different XML Schema trees is distributed by their semantic goals, and therefore, their appearance in XML tree is not equal.

Other related works [9, 10] proposed the method to measure the similarity between two XML Schemas, but their methodology only focused on the structural computations.

There are some approaches that mention to measure the semantic identity between elements of the two schemas. Do et al. [5] proposed a method to compute the similarity among names of two elements by using string matching. However, they also concentrated on the name similarity between concepts and did not explore the datatype and cardinality constraint of the XML Schema's elements. Another approach introduced the method based on linguistic taxonomy, such as [6, 8]. On the basis of concept definitions in WordNet, they can gain the most accurate semantics for words in the element names. WordNet is the large lexical database that is applied in various applications, such as in Web search [15], and Web query expansion [16]. In our methodology, we also use WordNet to measure the element name similarity.

There are some approaches that have been developed to find the structure similarity between two elements. The traditional approach was based on the information content [17]. These approaches computed the resemblance between two elements x and y based on the amount of information needed to describe the commonality between them. The value of similarity is high if there are more descriptions for (x, y). According to the information theory, the more specific the element, the more information is required to

portray them. Then, the degree of specificity is computed by the information content. The information content was also applied to the semantic relatedness of word senses in WordNet [18].

From the above related works, we can see that there is no complete approach to measure both semantic and structure similarity of two elements in two schemas. Each of the existing approach has its strength and weakness. Therefore, it is better if we can combine the strength of these methods to improve the accuracy in measuring the similarity of two schemas. That is the reason why we choose the method of calculating both semantic and structure resemblance of elements in two schemas.

Our measurement method is a little similar to the approach proposed in [7]. It calculated the similarity between two DTD trees by clustering the DTD's elements. The clustering is a technique that allows similar information to be stored together; thus, the more criteria for classifying the information, the greater quality of the clustering data. For instance, Ye et al. [19] added the weight factor in their clustering method. However, XClust [7] mainly relied on the user's judgment and the structural information to clustering the DTD elements. Moreover, XClust did not compute the datatype similarity of the attribute types inside DTD. Datatype is also an important factor contributing the quality of the element similarity.

We take this idea of clustering DTD data and then apply for XML Schema data. Moreover, we propose the new technique for measuring the datatype and constraint of two elements.

3 Semantic and structure similarity measurement

In this step, we explain our new matching mechanism between two XML Schemas. In addition to having similar characteristics, our solution has the following properties:

- It allows automated linguistic-based matching.
- It is both element-based and structured-based.
- It is biased toward similarity of leaf elements, where much schema semantics is captured.
- It exploits the internal structure, but is not overly misled by variations in that structure.

To illustrate for our method, we first restrict ourselves to hierarchical schemas. Thus, we model the interconnected elements of an XML Schema as a schema tree. We use two XML Schema trees below for explaining our algorithm.

We would like to match the two XML Schemas, *patient* in the Fig. 1a and b. The schemas are encoded as graphs, where the nodes represent schema elements. Although even a casual detector can see that both schemas are quite similar, there is still much variation in naming and structuring that make the matching algorithm being challenged.

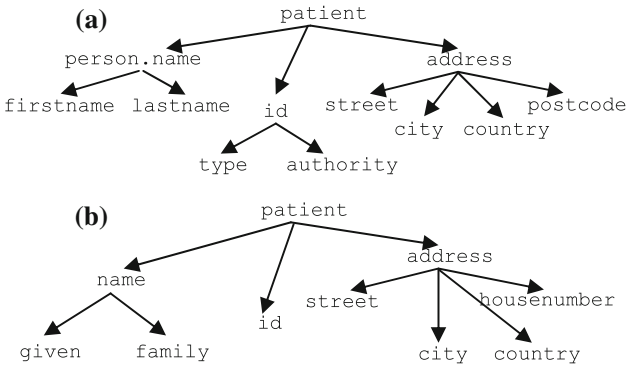

Fig. 1 Two different schema trees of *patient*

Table 1 Datatype compatibility table

	String	Date	Decimal	Integer	Float	Language
String	*1.00*	0.25	0.25	0.25	0.25	0.50
Date	0.25	*1.00*	0.58	0.58	0.58	0.25
Decimal	0.25	0.58	*1.00*	0.75	0.58	0.25
Integer	0.25	0.58	0.75	*1.00*	0.58	0.25
Float	0.25	0.58	0.58	0.58	*1.00*	0.25
Language	0.50	0.25	0.25	0.25	0.25	*1.00*

Similar to the previous works [5, 7, 18], we compute the similarity coefficients between elements of the two schemas and then deducting a mapping from those coefficients. The coefficients are calculated in two stages. The first stage, *semantic* matching, compares individual elements based on their names (linguistics), datatype, and cardinality constraint similarities. The similarity among datatypes is given in the Table 1. The constraint similarity value is drawn from the Table 2 and *CSim* metric. To handle the abbreviation of names (linguistic similarity), we use the WordNet [20] to determine whether these names are synonym or not. Followings are the detail of each similarity measurement.

3.1 Semantic similarity measurement

The semantics of a concept plays an important role in integrating the text documents [21]. Semantics of XML Schema comprises the vocabularies, the content model, and the datatype. Usually, XML Schema uses the standard namespace (*xs* or *xsd*), and the URI associated with this namespace to begin the document. With XML Schema, we can define the number of possible occurrences for an element with the *maxOccurs* and *minOccurs* attributes. Moreover, *simpleType* or *complexType* element helps us to differentiate the datatype similarity between two attribute

Table 2 Cardinality constraint similarity table

	Min = 0, max = unbound	Min = 1, max = unbound	Min = 0, max = 1	Min = 1, max = 1
Min = 0, max = unbound	*1.00*	0.5	0.67	0.17
Min = 1, max = unbound	0.5	*1.00*	0.17	0.67
Min = 0, max = 1	0.67	0.17	*1.00*	0.5
Min = 1, max = 1	0.17	0.67	0.5	*1.00*

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema
xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="patient">
  <xs:complexType> <xs:sequence>
     <xs:element ref="person.name" />
     <xs:element ref="id"
                    maxOccurs="unbounded"/>
        <xs:element ref="address" />
      </xs:sequence> </xs:complexType>
</xs:element>
<xs:element name="person.name">
    <xs:complexType> <xs:choice>
        <xs:element ref="firstname" />
        <xs:element ref="lastname" />
      </xs:choice> </xs:complexType>
  </xs:element>
 <xs:element name="id">
    <xs:complexType mixed="true">
      <xs:attribute name="type"
          type="xs:NMTOKEN" use="optional" />
      <xs:attribute name="authority"
          type="xs:NMTOKEN" use="optional" />
    </xs:complexType></xs:element>...
</xs:schema>
```

Fig. 2 XML Schema for *patient*

types of the elements. For instance, the datatype similarity between simple and complex elements is zero. Figure 2 presents an example of an XML Schema for *patient*.

3.1.1 Datatype similarity

Although the main factor for semantic similarity calculation is the element name, the consideration for other components also plays a very important role. For instance, the name similarity between two elements *id* in Fig. 1a and b is 1. However, this is a false matched value, since the first *id* element is a complex element, whereas the second *id* element is a simple element. This means they are different in other attributes. Therefore, it is necessary to use other factors to calculate their semantic relatedness to eliminate some false matches.

In the XML Schema document, every element is always either simple or complex type. If two elements have the same name and their datatype properties are identical (both are a complex type or simple type), their semantic similarity may be higher than other cases, such as simple and complex. Since the complex element contains children, in order to compute the similarity between two complex elements, we have to compare the similarity of their children. This problem is mentioned in the structure similarity measurement section. For two leaf elements, we concern their datatype declaration. Since the datatype often comes with an attribute element, the datatype measurement is only applied for the case that both elements are attributes. In the case that two elements are complex types or the first element is a complex type, the second is the simple type, the datatype similarity is 0. For example, the element *id* in the Fig. 1a has a complex type, and element *id* in the Fig. 1b has a string datatype. Therefore, the datatype similarity between two elements (complex, string) is 0.

Difference with the method proposed in [11], in which datatype similarity values are drawn by user's judgment, we propose a new metric to measure this relation. Particularly, we explore the inside characteristics of each datatype and compare the relatedness between them. On the basis of the summary about XML Schema datatype [22], we propose the technique to evaluate the datatype similarity based on the constraining facets of each datatype. It's named *DSim*, and it is determined by following equation:

$$DSim(d_1, d_2) = \frac{\sum_i \left|\left\{cf_i | d_1[cf_i] = d_2[cf_i], 1 \le i \le n_{cf}\right\}\right|}{n_{cf}} \quad (1)$$

where d_1 and d_2 are arbitrary datatypes presented in Table 1; *cf* is the list of datatype's constraining facets. It includes *length*, *minLength*, *maxLength*, *pattern*, *enumeration*, *whiteSpace*, *maxInclusive*, *minInclusive*, *maxExclusive*, *minExclusive*, *totalDigits*, and *fractionDigits*; and n_{cf} is the number of constraining facets, in this case $n_{cf} = 12$.

Table 1 presents the datatype similarity results of six typical attribute types in the XML Schema. Values in this table are computed based on the Eq. (1).

In the Table 1, *integer* and *decimal* have more similarity than with other types. When two elements have the same data type, their similarity has the highest (1.0, shown in italics).

3.1.2 Constraint similarity

Another factor that affects the semantic similarity between two elements is the cardinality (occurrence) constraint. It is

declared as *minOccurs* and *maxOccurs* in the XML Schema document. The *minOccurs* and *maxOccurs,* respectively, define the minimum and maximum number of occurrence times of an element in XML instances.

We use $CSim(d_1,d_2)$ to specify the constraint similarity between two elements d_1 and d_2. Different to the constraint table proposed in [11, 17], in which values are decided by human judgment, we define a novel metric to compute the constraint similarity values. For the definitely values of *minOccurs* and *maxOccurs*, we use the following equation for computing their cardinality constraint similarity:

$$CSim(e_1(\min,\max),e_2(\min,\max)) = \frac{\left(1-\frac{|e_1.\min-e_2.\min|}{e_1.\min+e_2.\min}\right)+\left(1-\frac{|e_1.\max-e_2.\max|}{e_1.\max+e_2.\max}\right)}{2} \quad (2)$$

In (2), *min* and *max* are short forms of *minOccurs* and *maxOccurs*, respectively. Usually, *minOccurs* is assigned by 0 or 1, and *maxOccurs* is 1 or *unbound*. Usually, the value of *maxOccurs* is undetermined (*unbound*). To measure the *CSim* for this value, we use the following function:

$$d_1[maxOccurs = Unbound] = 5 * MAX(d_2[maxOccurs]) \quad (3)$$

We decide to use this function, since we have surveyed in our dataset (XSD and XML instances), the appearance time of an attribute with $maxOccurs = Unbound$ is about five times greater than maximum values of definitely *maxOccurs*. In the case that all *maxOccurs* in the XSD document are undetermined, we assign their values as 5. Taking this value and then apply for the Eq. (2), we harvest the similarity of the attribute's cardinality constraint. The details are presented in Table 2.

Table 2 presents the cardinality constraint similarity when the value of *unbound* is 5. The constraint similarity has its highest (1.0, shown in italics) when two elements have the same cardinality constraint.

Sometimes, the values of the *minOccurs* and *maxOccurs* are retrieved from other indicators. For instance, *Order* and *Group* indicators (*any*, *all*, *choice*, *sequence*, *group name*, and *group reference*) have the default value for *maxOccurs* and *minOccurs* is 1.

3.1.3 Name similarity

The most important factor for the semantic measurement is the linguistic similarity of two elements.

To determine the linguistic similarity between elements, we use the algorithm that is presented in Fig. 3. Specifically, the algorithm finds the similarity between two elements e_1 and e_2. The breadth-first search is executed

Algorithm: **Name Similarity**
Input: Two elements, e and e'
Distance = 5 – level;
Output: Name similarity
if $e_1.name==e_2.name$ **then return 1; else return** *DepthSyn*(e_1, {e_2}, *1evel);*
{e_2}=*S;*
Function *DepthSyn(e, S, level)*
Output: the synonym in depth
if (level ≥ distance) **then return 0;**
else if ($e_1 \in S$) **then return** *power(0.9,distance)*;
return *DepthSyn(e,S, distance+1);*

Fig. 3 The linguistic similarity algorithm

starting from the synonym set on WordNet of the element e_1 to the synonym set of the element e_2, and so on, until e_2 is matched. If the target is not found, then the linguistic similarity returns value 0, otherwise it is calculated as 0.9^{distance}.

Definition 1: *Semantic similarity* captures the similarity between the names, constraints, and path context of two elements. This is given by:

$$SeSim(e_1,e_2) = \alpha * NameSim(e_1,e_2) + \beta * DSim(d_1,d_2) + (1-\alpha-\beta) * CSim(e_1,e_2) \quad (4)$$

where *SeSim* is the semantic similarity; α and β are the weighted constants by the program. In our experiments, we assign $\alpha = 0.32$ and $\beta = 0.34$; *NameSim* is the name similarity calculated by the algorithm in Fig. 3; *DSim* is the similarity between two datatypes d_1 and d_2, and *CSim* is the cardinality constraint similarity of e_1 and e_2 elements.

The *level* in Fig. 3 is the depth of element in the XML Schema tree. We make a default the depth of the root element is 0, and its direct children depth is 1 and so on. For example, in the Fig. 1a, the *element level* of the element *person.name* is 1, that of the element *lastname* is 2, etc.

3.2 Structure similarity measurement

The second stage is called *structure* matching. It matches the schema elements based on the similarity of their context (position) and their nearest elements. For instance, *homenumber* in Fig. 1b is mapped to *postcode* in Fig. 1a, since their parent names, *address*, are same, and the other three children, *street*, *city*, *country*, are also matched to each others.

The structure matching depends in part on the semantic similarity that is computed in the first stage. For instance, *given* and *family* in Fig. 1b should match to *firstname* and

lastname in Fig. 1a, rather than to the *type* and *authority* under element *id*, since element *id* in Fig. 1a already matches to the same element in Fig. 1b. The result is a structure similarity coefficient, *StSim*, for each pair of elements.

The context of an element is composed of ancestor, sibling, immediate children and leaves. Two elements have a structural similarity if they are similar in contexts. In our algorithm, the structure similarity is computed based on the following principles:

- Elements that are leaves of the two trees are similar if their tags are similar, and the elements in their ancestors and siblings are similar.
- Two non-leaf elements are similar if their tags are similar, and the sub-tree rooted at the two elements are similar; and
- Two non-leaf elements are structurally similar if their leaf sets are highly similar, even if their immediate children are not.

The structure similarity measurement of two schema trees is presented in Fig. 4.

In Fig. 4, we choose the minimum value of threshold is 0 and maximum is 0.3. Every time of comparing one of two values is changed by 0.1. The maximum threshold can be altered. Because in the first stage of computing semantic similarity, we select the maximum element depth 3, so it is same with structural measurement.

In contrary to the semantic computation, the depth-first search order, the structural relatedness computation visits the element from the leaf node to the root node, following post-order traversal algorithm.

The elements in the two trees are then enumerated in post-order, which is uniquely defined for a given tree.

Algorithm: **Tree_similarity;** ***//It is based on the structure similarity among pair of elements,*** **StSim**
Input: Two schema trees S, and T
thresh_min=0; thresh_max=0.3
Output: The structure similarity
for each s∈ S, t∈ T where s, t are leaves
S'=post-order(S); T'=post-order(T);
for each s in S',
  for each t in T' t∈ T
    if StSim(s,t) ≥ thresh_max **then**
      StSim(s,t)=StSim(s,t)+0.1;
    else if StSim(s,t) ≤ thresh_min **then**
      StSim(s,t)=StSim(s,t)-0.1;
Tree_Similarity(S,T)=StSim(s,t);

Fig. 4 The structure similarity algorithm

The first step in the loop calculates the structure similarity between two elements. For leaves, this is just the value of *StSim* that is calculated in the previous step. When one of the two elements is not a leaf, the structural similarity is calculated as a measure of the number of leaf level matches in the sub-trees.

In order to compute the similarity between two schema trees, we have to complete the structure similarity between each pair of elements in two trees.

For each pair of schema elements, the algorithm compares the structural similarity, *StSim*. It is the similarity of the contexts in which the elements occur in the two schemas.

We say that a leaf in the first schema has a link to another leaf in the second schema is their linguistic similarity, *LingSim*, exceeds the *thresh_max*. We assume that condition since the structure of a document is associated with the semantic similarity of their elements, too. If we do not consider the semantic similarity, it may lead to the case that two schema trees have the same structure, but they are different in element names. We choose the low value of *thresh_max* in order to increase the value of structure similarity measurement. The structure similarity is defined in the definition 2.

Definition 2: The *structure similarity* between two elements e_1 and e_2 is specified as:

$$StSim(e_1, e_2) = \frac{sum\_links(e_1, e_2) + sum\_links(e_2, e_1)}{leaves(e_1) + leaves(e_2)} \tag{5}$$

where $leaves(e_1)$ is the total number of leaves in the sub-tree rooted at element e_1; $sum\_links(e_1, e_2)$ is the total number of links from the leaves of element e_1 to the leaves of element e_2.

For example, let us compute the structural similarity of two elements, *address* in Fig. 1a and b.

$$\begin{aligned} leaves(address) + leaves(address\prime) &= 4 + 4 = 8 \\ sum\_link(address, address\prime) &= 3 \\ sum\_link(address\prime, address) &= 3 \\ \Rightarrow StSim(address, address\prime) &= 6/8 = 0.75 \end{aligned}$$

It means that if the semantic similarity between each pair of leaves in the two sub-trees is greater than *thresh_max*, then the *sum_link* increases by one (i.e. $sum\_links = sum\_links + 1$). Otherwise, if its semantics is less than *thresh_max*, the *sum_link* is decreased by one (i.e. $sum\_links = sum\_links - 1$).

3.3 Element similarity measurement

Our approach considers both structure and semantics of elements in the tree. Therefore, the similarity between two

elements in both XML Schema graphs is calculated as the weighted sum of these two components:

$$ESim(s,t) = \delta * SeSim(s,t) + (1-\delta) * StSim(s,t) \quad (6)$$

where δ is the weighted value, $0 < \delta \leq 1$.

To understand our algorithm, let us compute the element similarity of some pairs of elements given in Fig. 1. To distinguish between elements with the name labels in two schemas, we put an apostrophe in the element name of the second schema.

For example, we compute the element similarity of the elements *person.name* and *name* in Fig. 1a and b, respectively. Because two elements have differences in the name, we have to compute their linguistic similarity by using WordNet [8]. The element *person.name* is tokenized as *person* and *name*. The last token is matched with an element *name* in Fig. 1a. Therefore, the linguistic similarity of *person.name* and *name* is 0.8. Since both elements are complex types, their datatype similarity is 0. Moreover, their cardinality constraint value is same, so the constraint similarity is 1.

For example, if we are interested in the linguistic similarity, we assign values for $\alpha = 0.6$, $\beta = 0.2$

$$SeSim(person.name, name) = 0.6 * 0.8 + 0.2 = 0.68$$

Now, we compute their structure similarity.

$$leaves(person.name) + leaves(name) = 2 + 2 = 4$$
$$sum\_links(person.name, name) = 2$$
$$sum\_links(name, person.name) = 2$$

Therefore, $StSim(person.name, name) = 4/4 = 1$.

If we place balance the weight on both semantic and structural similarity, $\delta = 0.5$, we have:

$$ESim(person.name, name) = 0.5 * 0.68 + 0.5 * 1 = 0.84$$

When we have the element similarity of all element pairs in two schemas, we can compute the similarity of the two schemas. The similarity between two XML Schemas trees is calculated as the weighted sum of two components:

$$ScheSim(T_1, T_2) = \varepsilon * \sum_{i=1}^{k} SeSim(e_1, e_2) + (1-\varepsilon) * TreeSim(T_1, T_2) \quad (7)$$

where *ScheSim* is the schema tree similarity; ε is the weighted value, $0 < \varepsilon \leq 1$; k is the lowest number of elements in the tree, for example, if the schema T_1 has total 230 elements, schema T_2 has 220 elements, then $k = 220$; e_1 and e_2 are the elements of T_1 and T_2, respectively; *TreeSim* is the tree similarity of two schemas.

The similarity calculation has a close relationship to each other and a recursion. Two elements are semantic similar if their leaf sets are similar. The semantic similarity of the leaves is increases if their ancestors are highly similar. The similarity of the structure is also influenced by the semantic similarity. If the sub-tree of two elements are high similar, the structure similarity of their ancestor is high, too.

4 Experimental results

The element similarity including semantic and structural similarity is implemented by C# language. To examine the performance of ESim, we use more than 20 healthcare XML Schemas (and corresponding DTDs) from [23, 24]. We compare our method with the most related work (XClust-the similarity between two DTD trees) and the similarity between XML documents (called *XMLSim*) [18]. For medical XML documents without schema files, we draw XML Schemas and DTD from the XML instances by using the HITSoftware.[1] Results from using various similarity measures were obtained and reported in Table 3 below.

The weighting factors used in this experiment are $\alpha = 0.32$, $\beta = 0.34$, $\delta = 0.6$, and $\varepsilon = 0.6$.

As shown in the table, *XMLSim* has a lowest matching. This is because the number of instances in the XML documents is variable and is not same with each other document. When one node of the first document is matched with another node in the second document, the algorithm computes for the next pair of nodes. Thus, if the number of nodes in the two documents is different, the matched value is decreased.

In order to answer the question of how much the information gets lost when matching two XML Schemas, we use the metrics proposed by Do Hong-Hai [25]. The evaluations use the following calculations.

Recall: "It specifies the share of real correspondences that is found" [25].

$$recall = \frac{found\_proposed\_correspondences}{all\_proposed\_correspondences} \quad (8)$$

Precision: "It reflects the share of real correspondences among all found correspondences" [25].

Table 3 Experiment results

Similarity measure	Number of element pairs	Number matches
XClust	41	34
XMLSim	41	27
ESim	41	37

[1] http://www.hitsw.com/xml_utilites/.

$$precision = \frac{found\_proposed\_correspondences}{all\_found\_correspondences} \quad (9)$$

Precision expresses the accuracy of the matching. In order to get a more significant statement on matchers' quality, the *F_measure* formula is introduced [25].

$$F\_measure = 2 * \frac{precision * recall}{precision + recall} \quad (10)$$

Finally, the metric for calculating the schema matching is known as "overall." It is defined as follows:

$$overall = recall * \left(2 - \left(\frac{1}{precision}\right)\right) \quad (11)$$

The calculation results for our methodology (*ESim*), *XMLSim* [18], and *XClust* [7] are illustrated in Fig. 5.

Figure 5 shows that our matching quality is higher than those of *XMLSim* and *XCLust*'s methods. The reason is that the element similarity measurement in *XClust* did not concern the datatype similarity between two elements, whereas some element pairs have the same name, but they are different in datatypes. The *XMLSim* paid too much attention to the information content similarity and did not mention about the datatype as well as the cardinality constraint similarity of two elements. Therefore, it gets lowest values in all calculations.

In our experiments, the threshold values are chosen between 0.3 and 1. In order to evaluate the effect of each measuring factor to the computation results, we conduct a set of experiments calculating for each factor only and then compare their *F_measure* values with the *F_measure*'s ESim.

Figures 6, 7, 8, and 9 show the measuring quality of the name, datatype, cardinality constraint, and structure of the elements in XSD healthcare documents. These figures indicate that there is no single measuring factor is able to determine the good correspondences.

Figure 10 indicates that the combination of all factors gives the highest *F_measure* values, especially when we increase the threshold values. Among those factors, the structure calculation (*StSim*) produces the greatest *F_measure* values and following by the name

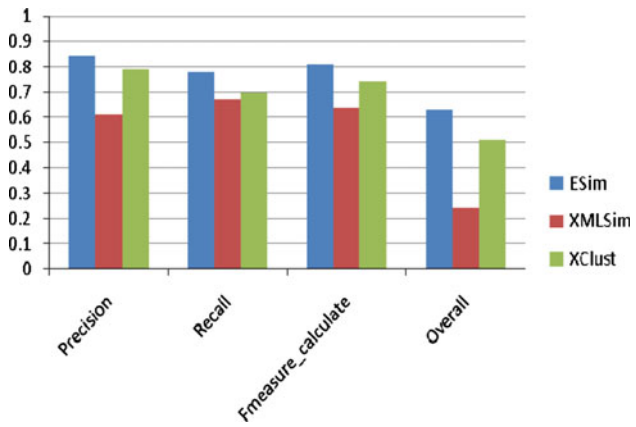

Fig. 5 Matching comparisons of our method (ESim) to XMLSim [6] and XClust [17]

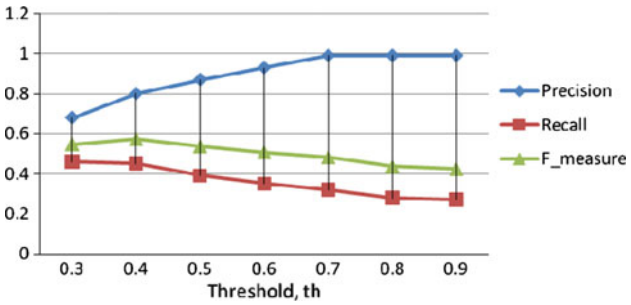

Fig. 6 Quality of name measure

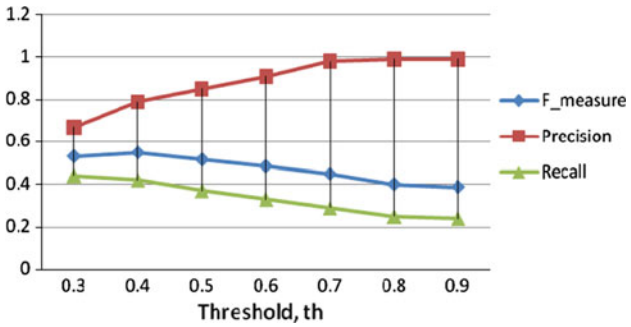

Fig. 7 Quality of datatype measure

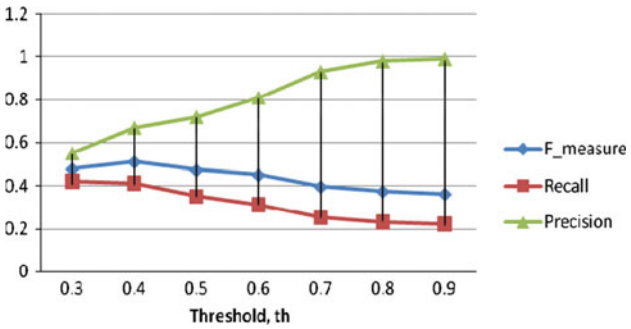

Fig. 8 Quality of cardinality constraint measure

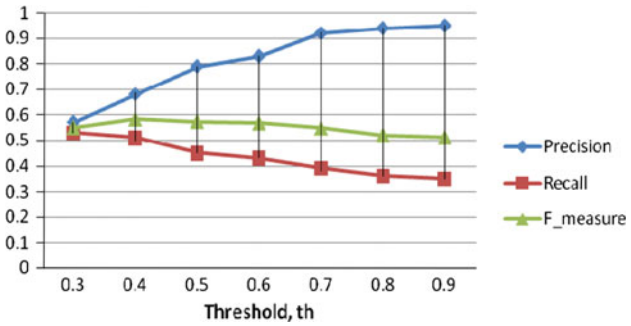

Fig. 9 Quality of structure measure

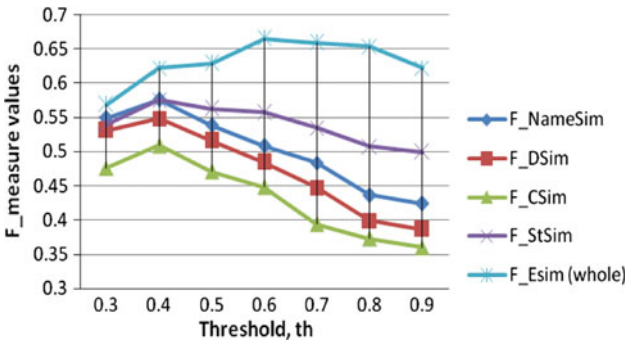

Fig. 10 *F_measure* comparison

measurement, datatype calculation, and the cardinality constraint estimation.

The experimental results conducted in this study show that although the structural resemblance plays an important role in the measure, its combination with other semantic factors produces the best matches.

5 Conclusions and future work

In this paper, we propose an innovative semantic and structure similarity approach for both XML healthcare Schemas. We provide novel metrics to compute the datatype similarity for attribute types and measure the cardinality constraint resemblance of two elements. We also present a new algorithm in computing the element similarity that improves the past methods in many respects. For instance, the semantic similarity between two elements is included not only their linguistic similarity but also their datatype and constraint compatibilities. The structure similarity mentions both the distance between two elements and their linguistic meaning. To evaluate these measures, we conduct a set of experiments to compare our method with the related works and the important role of each measuring factor. This exposes the strength of our approach and is a possible method for future comparisons.

Our long-term goal is to measure the similarities of unstructured healthcare data and apply these algorithms to compare the similarity of different models, such as XML and OWL ontology.

Acknowledgments This work was supported by a grant from the Kyung Hee University in 2010 (KHU-20101372).

References

1. Brown I, Adams A (2007) The ethical challenges of ubiquitous healthcare. Int Rev Inf Ethics 8(12):53–60
2. Wikipedia, Electronic healthcare record, http://en.wikipedia.org/wiki/Electronic_health_record
3. Jervis M (2002) XML DTDs vs XML Schema. http://www.sitepoint.com/xml-dtds-xml-schema/
4. Wu Z, Palmer M (1994) Verbs semantics and lexical selection. In: Proceedings of the 32nd annual meeting on association for computational linguistics, pp 133–138
5. Do H-H, Rahm E (2002) COMA—a system for flexible combination of schema matching approaches. In: Proceedings of the very large data bases conference (VLDB), pp 610–621
6. Yang DD, David MW (2005) Powers, measuring semantic similarity in the taxonomy of WordNet. The 28th Australasian computer science conference (ACSC2005), pp 315–322
7. Lee ML, Yang LH, Hsu W, Yang X (2002) XCLust: clustering XML schemas for effective integration. ACM Press, New York, pp 292–299
8. Princeton University, WordNet_ A lexical database for English, http://wordnet.princeton.edu/wordnet
9. Tekli J, Chbeir R, Yetongnon K (2007) A hydrid approach for XML similarity. In: SOFSEM '07 proceedings of the 33rd conference on current trends in theory and practice of computer science. Springer, Berlin, pp783–795
10. Tekli J, Chbeir R, Yetongnon K (2009) An overview on XML similarity: background, current trends and future directions. Comput Sci Rev 3:151–173
11. Algergawy A, Nayak R, Saake G (2010) Element similarity measures in XML schema matching. Inf Sci 180:4975–4998
12. Rada R, Mili H, Bicknell E, Blettner M (1989) Development and application of a metric on semantic nets. IEEE Trans Syst Man Cybern 19(1):17–30
13. Fernandez A, Polleres A, Ossowski S (2007) Towards fine-grained service matchmaking by using concept similarity. In: Workshop on service matchmaking and resource retrieval in the semantic web, pp 31–45
14. Li Y, Bandar Z, McLean D (2003) An approach for measuring semantic similarity between words using multiple information sources. IEEE Trans Knowl Data Eng 15(4):871–882
15. Pyshkin E, Kuznetsov A (2010) Approaches for web search user interfaces: how to improve the search quality for various types of information. J Conver 1(1):1–8
16. Klyuev V, Yokoyama A (2010) Web query expansion: a strategy utilising Japanese WordNet. J Conver 1(1):23–28
17. Lin D (1998) An information-theoretic definition of similarity. In: Proceedings of international conference on machine learning, pp 296–304
18. Resnik P (1999) Semantic similarity in a taxonomy an information-based measure and its applications to problems of ambiguity in natural language. J Artif Intell Res 11:95–130
19. Ye Y, Li X, Wu B, Li Y (2011) A comparative study of feature weighting methods for document co-clustering. IJITCC 1(2):206–220
20. Leacock C, Chodorow M (1998) Combining local context and WordNet similarity for word sense identification. In: Fellbaum C (ed) WordNet: an electronic lexical database. MIT Press, Cambridge, pp 265–283
21. Klyuev V, Oleshchuk V (2011) Semantic retrieval: an approach to representing, searching and summarizing text documents. IJITCC 1(2):221–234
22. D Vint Productions (2003) XML schema—data types quick reference. http://www.xml.dvint.com
23. Mebiquitous XML Schema. http://ns.medbiq.org/
24. Health Level Seven International. http://www.hl7.org/
25. Do H–H (2005) Schema matching and mapping-based data integration, PhD thesis, University of Leipzig, Interdisciplinary Center for Bioinformatics and Department of Computer Science