Measuring UML Model Similarity

Jie Su and Junpeng Bao
Department of Computer Science & Technology, Xi'an Jiaotong University, Xi'an 710049, P.R. China

Keywords: UML, XML, Model Similarity Edit Distance Edit.

Abstract: Many user requirements and UML models are similar even if identical, but their application backgrounds are different. It is a straight and feasible way to mine those similar UML models for a model warehouse and reuse them so as to improve software development efficiency. The key point in the idea is to measure the similarity of UML models. We present a Level Edit Distance method to solve the problem. The LED measures similarity of XML structures instead of UML models. Indeed, UML models are converted to XML documents according to XMI so that UML model similarity equals to XML document similarity. However, our method concentrates on the pure structural similarity of UML models in XMI format, namely, the semantic information is ignored. The LED is different from the traditional Edit Distance. The former needs only one primitive operation whereas the later needs three. Our preparatory experimental results show that the LED can keep almost the same distance distribution with the traditional ED and is a little faster than the latter. We are going to improve the capability of the LED and combine it with a semantic-considered method in order to precisely evaluate the similarity of user requirements.

1 INTRODUCTION

UML (Unified Modeling Language) is a standard general-purpose modeling language in the area of object-oriented software engineering. As a result, more and more software engineering documents describe user requirements and build models in UML. UML models are basic foundation in the Model Driven Development. It is a fact that many software developers rebuild a variety of models and functions again and again. Obviously, a lot of rebuilt codes are not better than the existed. One reason of the low reusability is that it is too hard or time consuming to find the most appropriate model from a huge mountain of boring model repository. Whatsoever, many user requirements and UML models are similar but their application backgrounds are different. That means their schema and structure are similar even if identical. Hence, a straight way to improve the model reusability and software development efficiency is that to mine the similar UML models for a model warehouse and reuse them as far as possible. Our ongoing research aims to develop an effective method to measuring UML model similarity so as to improve the reusability of them.

Instead of straight measuring similarity of UML models, we exploit XMI (XML Metadata Interchange) to transfer a UML model into a XML document so that the similarity of UML models equals to that of XML texts. A XML text is a semi-structured text that organizes its content in a tree structure with labeled tags. The similarity or identity of two XML texts means that (1) the text contents are similar or identical between them; (2) their structures are also similar or identical to each other. It has to be noted that a XML text is different form a plain text because the former has explicit tree structure whereas the latter has not. One of key points in the XML similarity is the structural similarity of XML texts. Since XML becomes a basic standard document format in the World Wide Web. There are many XML structural similarity measure approaches. The Edit Distance based methods are a straight way to measure structural similarity of XML texts. Another way is to find similar paths or sub-trees in two XML texts, and then calculate the portion of similar paths or sub-trees out of the whole structure tree. The serial sequence methods convert the tree structure of a semi-structured text into a sequence of code, and then calculate the similarity of code sequences to assess the similarity of semi-structured texts.

Wen (L. Wen, etc. 2008) presented a XML structural similarity measure method, which is typical traditional Edit Distance based method. They convert the tree structure into a node sequence

Su J. and Bao J..
Measuring UML Model Similarity.
DOI: 10.5220/0004027303190323
In *Proceedings of the 7th International Conference on Software Paradigm Trends* (ICSOFT-2012), pages 319-323
ISBN: 978-989-8565-19-8

which contains tag and left bracket, and then extract subsequences. They believe that the similarity of sub-sequences is equal to that of the XML structure.

In this paper, we concentrate on the problem of pure structural similarity of UML models in XMI format. That means the tag semantic information is discarded. We present a modified Edit Distance method, called Level Edit Distance (LED), to calculate the similarity of two tree structures based on only 1 primitive operation. Whereas the traditional edit distance needs 3 primitive operations, including change, insertion and deletion. Additionally, LED calculates level distance at each level and then sums them up with different weight to get the final distance between trees. But Wen calculates distance only on the sub-sequences.

2 MEASURE OF STRUCTURAL SIMILARITY

2.1 Similarity Principles

An UML model can be exactly converted into a XMI document, i.e. a well formed XML text so that the similarity of UML models equals to that of XML texts. A XML text contains both structural information and semantic information. But in this paper, we consider only structure information, ignore semantic information. Because we aim to find more similar models in a repository and try to improve the reusability of various models. Obviously, the high level model and general model contains less semantic information so as to guarantee their application to different backgrounds. Additionally, they are more abstract and comply with platform independent philosophy. In terms of reusability, semantic information tends to obstruct the model similarity. In fact, the abstract structures of UML models in different application backgrounds may be very similar even if identical. The most differences in various user requirements are often from semantic narrative texts. So we omit a model's semantic information and remain its structure information in order to compare models in a higher and more abstract level. We believe it is helpful to improve the reusability of models. As a result, the method proposed in this paper concentrates on the pure structural similarity of UML models in XMI format. Indeed, it is the similarity of trees.

The Fig. 1 shows 4 trees extracted from 4 XML texts. However, the similarities among each other of them are depended on our subject definition. There are different principles so that the similarity relationships among the 4 trees are various.

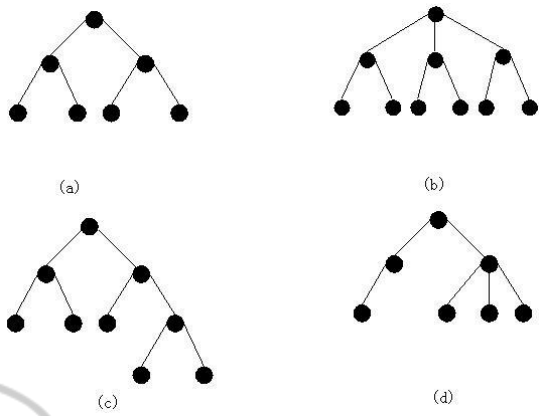

Figure 1: Pure structure of 4 XML documents.

According to the traversing sequence in a tree, we define two principles to decide the relationships among the tree structural similarities.

Principle 1: To compare two trees in deep first way. Namely, two trees are compared from left to right. For example, in the Fig. 1, according to this principle, the similarity between tree a and b is greater than that between a and d. Also, the similarity between tree a and c is greather than that between a and d, i.e.

$sim(a,b) > sim(a,d)$ and $sim(a,c) > sim(a,d)$

Where sim(a,b) means the structural similarity between tree a and b.

Principle 2: To compare two trees in broad first way. Namely, two trees are compared from up to bottom. It implies that the more differences at the lower level, the more differences between two trees. The root of a tree is level 0. According to this principle, the relationships among the structural similarities are:

$sim(a,c) > sim(a,d) > sim(a,b)$

Because the tree a and c are still identical at the level 2 but tree a is different from d at that level. Tree a and b are different from the level 1.

In this paper our method complies with the principle 2, and the lower level will has a great effect on the whole similarity. Indeed a 10 based level weight is attached to each level in our method.

2.2 Similarity Algorithm

Wen's method is a typical traditional Edit Distance, which is the minimum operation cost to transform one string to another with three primitive operations including change, insertion and deletion.

Given two XML document d_1 and d_2, the ED of Wen's method is defined as follows.

Table 1: The node sequences of 4 trees after null nodes inserted.

Tree	Level 0	Level 1			Level 2							Level 3	
(a)	N_0	$N_{0.1}$	$N_{0.2}$	Null	$N_{0.1.1}$	$N_{0.1.2}$	$N_{0.2.1}$	$N_{0.2.2}$	Null	Null	Null	Null	Null
(b)	N_0	$N_{0.1}$	$N_{0.2}$	$N_{0.3}$	$N_{0.1.1}$	$N_{0.1.2}$	$N_{0.2.1}$	$N_{0.2.2}$	Null	$N_{0.3.1}$	$N_{0.3.2}$	Null	Null
(c)	N_0	$N_{0.1}$	$N_{0.2}$	Null	$N_{0.1.1}$	$N_{0.1.2}$	$N_{0.2.1}$	$N_{0.2.2}$	Null	Null	Null	$N_{0.2.2.1}$	$N_{0.2.2.2}$
(d)	N_0	$N_{0.1}$	$N_{0.2}$	Null	$N_{0.1.1}$	Null	$N_{0.2.1}$	$N_{0.2.2}$	$N_{0.2.3}$	Null	Null	Null	Null

Table 2: The final code sequences of 4 trees.

Tree	Level 0	Level 1			Level 2							Level 3	
(a)	1	1	1	0	1	1	1	1	0	0	0	0	0
(b)	1	1	1	1	1	1	1	1	0	1	1	0	0
(c)	1	1	1	0	1	1	1	1	0	0	0	1	1
(d)	1	1	1	0	1	0	1	1	1	0	0	0	0

$$ED(d_1, d_2) = \frac{OPN(d_1, d_2)}{\max\{len(d_1), len(d_2)\}} \quad (1)$$

where OPN(d_1, d_2) denotes the minimum cost of operations to change the node sequence of d_1 to that of d_2. The len(d_1) denotes the length of node sequence of d_1.

But we think 1 primitive operation is enough to compare two strings. We present a Level Edit Distance method (LED) that needs one operation to change one string to another. We believe this method is more simple, understandable and easier to implement. The reduction of primitive operations has no effect on the structural similarity relationship. In the next section, the experimental results prove that LED and ED produce almost the same results.

The steps of measuring pure structural similarity of XMI by LED are:

(1) To traverse a XMI document structure in the Broad First way to produce a sequence of node with it's position code.

(2) To compare two node sequences at each identical
position, insert a node in a certain position if it is absent in one sequence.

(3) To encode the inserted node as 0, the original node as 1.

(4) To compute the distance between the codes of each level in two tree structures. The distance between codes of the *i* level in tree a and tree b is denoted by *LED(a,b,i)*, which is measured as follows,

$$\begin{aligned} &LED(a,b,i) = \sum_{f=1}^{m} H_f(a,b,i) \quad (i \geq 1) \\ &LED(a,b,0) = 0 \\ &m = \max\{N(a,i-1), N(b,i-1)\} \end{aligned} \quad (2)$$

where $N(a,i-1)$ the number of nodes at the level i-1 in the tree a. $H_f(a,b,i)$ denotes the Hamming distance between the codes that present at the level i and belong to the same father node at the level i-1. It implies that the same father node in two trees may contain different children nodes so that we exploit the Hamming distance to measure this difference. For example, the Fig. 1 shows four structures of XML documents. Their LEDs at each level are different from each other.

(5) To sum the LED at each level with level weight to get the final LED between two trees as follows.

$$LED(a,b) = \sum_{i=0}^{n} LED(a,b,i) \times 10^{-i} \quad (3)$$

where LED(a,b)denotes the final distance between tree a and tree b, i denotes the level of the tree structure, n denotes the max level between the tree a and tree b.

3 PREPARATORY EXPERIMENT

It is not easy to collect a large scale of the UML2.0 based materials. We download 150 UML model documents from the OMG Formal Specifications website* and select 50 XMI documents to test our idea. The LED value of each document pair is computed to get a LED based similarity matrix, which is compared with Wen's ED based similarity matrix.

The Figure 2 compares the ED and LED distances of a selected XMI document to all the other 49 XMI documents. The distance distributions

* http://www.omg.org/spec/

over 2 matrices are very similar except a few points, such as the 36th XMI pair in the figure 2. We check the XMI pair and find that it has a very big difference at some levels as shown in table3.

Table 3: The LED value of the 36th pair at each level.

level	0	1	2	3	4	5	6
LED	0	1	98	185	79	48	16

It is clear that the difference of the nodes in the two trees at the level 2 is 98 so that LED(a,b,2)×10^{-2}=0.98. That indeed leads to 0.9 increment at the level 1. In a similar way, the difference at the level 3 is 0.185, which leads to 0.1 increment at the level 1 and 0.08 at the level2. These excess increments disturb the final similarity evaluation. It implies that 10 is not the best level weight's base.

Table 4: Running time of LED and ED.

Number of XML	Total Size in MB	Running time in second	
		ED	*LED*
10	0.5713	2.6400	2.5469
50	1.52	35.6559	34.7809
100	4.45	216.9690	195.9379
200	10.42	808.546	809.9219
400	28.8	3359.5620	3330.3129

However, the LED has the same resolution ability with ED whereas LED needs only 1 primitive operation. In order to precisely evaluate the difference between LED and ED, we calculate the standard deviation of them as follows.

$$DE = \sqrt{\frac{1}{P}\sum_{i=1}^{P}(\Delta d_i - \overline{\Delta d})^2} \qquad (4)$$

Where P is the number of XML document pairs. $\Delta d_i = LED_i - ED_i$ denotes the difference between the LED of the ith XMI document pair and the ED of that pair. The $\overline{\Delta d}$ is the average of Δd_i. The standard deviation of the difference between LED and ED over 50 XMI documents is 0.2462.

In order to test the speed of the methods, the 50 XMI documents are duplicated many times to get the 100, 200 and 400 XMI test documents, as shown in the table4, which compares the running time of two methods. The fact is that LED is a little faster than ED.

4 CONCLUSIONS

In order to efficiently reuse amount of existed UML models and effectively evaluate the similarity of requirement documents, we present a modified edit distance method, called LED, to measure the similarity of UML models in XMI format. The main contributions of the LED are:

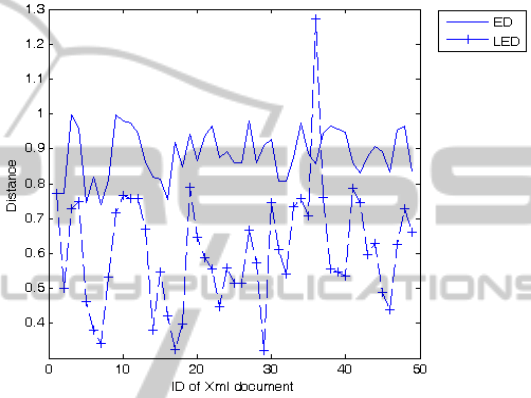

Figure 2: The ED and LED of the selected XML document to the others.

(1) LED needs only 1 primitive operation whereas the traditional edit distance needs 3 operations. As a result, the LED is easy to understand and program. Additionally, LED is a little faster than ED.

(2) LED ignores the tags' semantic so as to measure the pure structure similarity. In this way, it is easy to find the similar requirement in spite of different application background. That is to say, it is useful for developer to concentrate on upper level abstract model and promote reusability of existed models.

(3) LED considers the difference at each level and assigns greater weight to lower level. So it satisfies our up-down comparison intuition. The far small detail branches in trees have small effect on the similarity.

However, the LED method has a few defects that lead to a jump of the extreme wide layer. We will find a better level weight to solve the problem. Furthermore, the semantic information is crucial to finally fulfil the user requirements. LED can be used to find the appropriate models at the start of the development, not the whole life cycle. We are going to research a semantic-included XMI similarity measure model to solve the issue.

ACKNOWLEDGEMENTS

This research is supported by National Natural Science Foundation of China (Grant 60903123), the Fundamental Research Funds for the Central Universities and the Baidu Theme Research Plan on Large Scale Machine Learning and Data Mining.

REFERENCES

Yasser Kotb, 2010. Improving the UML Consistency Using Text Semantic Similarity Approach. *Proceedings of the 2nd International Conference on Computer Technology and Development*. Cairo, Egypt. pp. 90-94

Zhenchang Xing, Eleni Stroulia, 2005. Differencing logical UML models. *Proceedings of the 20th International Conference on Automated Software Engineering.* Volume 14, pp. 215-259

Udo Kelte, JürgenWehren, Jörg Niere, 2005. A Generic Difference Algorithm for UML Models. *Proceedings of Software Engineering. LNCS.*Volume 64, pp. 105-116

Joe Tekli, Richard Chbeir, Kokou Yetongnon, 2007. A Fine-Grained XML Structural Comparison Approach. *Proceedings of the 26th International Conference on Conceptual Modeling. LNCS.* Volume 4801, pp.582-598

Shihui Zheng, Aoying Zhou, Long Zhang, 2003. Similarity Measure and Structural Index of XML Documents. *Chinese Journal of computers.* 26(9):1116-1122

Tao Xie, Chaofeng Sha, Xiaoling Wang, Aoying Zhou, 2006. Approximate Top-k Structural Similarity Search. *Proceedings of the 8th Asia-Pacific Web Conference. LNCS.* Volume 3841, pp. 319-330

Guangming Xing, Jinhua Guo, Zhonghang Xia, 2007. Classifying XML Documents Based on Structure/Content Similarity. *Proceedings of the 5th International Workshop of the Initiative for the Evaluation of XML Retrieval. LNCS.* Volume 4518, pp. 444-457

Cuiming Lu, Fang Li, 2005. Simulation Research on XML Documents Similarity. *Chinese Simulation of Computers.* 22(12):300-303

H.J.Moon, J.W.Yoo, J.Choi, 2007. An Effective Detection Method for Clustering Similar XML DTDs Using Tag Sequences. *LNCS.* Volume 4706, pp. 849-860

Dunlu Peng, Huan Hou, Jing Lu, 2009. A Bloom Filter Based Approach for Evaluating Structural Similarity of XML Documents. *LNCS.* Volume 5854, pp. 242-251

W. Viyanon, S. K. Madria, 2009. A System for detecting XML Similarity in Content and Structure using Relational Database. *Proceedings of the 18th ACM Conference on Information and Knowledge Management.* pp. 197-1206

W. Viyanon, S.K. Madria, 2010. XML-SIM-CHANGE: Structure and Content Semantic Similarity Detection among XML Document Versions. *Proceedings of the Confederated International Conference On the Move to Meaningful Internet Systems. LNCS.* Volume 6427, pp. 1061-1078

Woosaeng Kim, 2008. XML document similarity measure in terms of the structure and contents. *Proceedings of the 2nd WSEAS International Conference on Computer Engineering and Applications.* pp. 205-212

L. Wen, T. Amagasa, H. Kitagawa, 2008. An Approach for XML Similarity Join Using Tree Serialization. *Proceedings of 13th International Conference on Database Systems for Advanced Applications. LNCS.* Volume 4947, pp. 562-570