An Automatic Ontology Matching Using Similarity of Components and Synonym

Akkarat Boonyapalanant[1]
and Supot Nitsuwat[2]
King Mongkut's University of Technology North Bangkok, Thailand
[1]akkarat.b@it.kmutnb.ac.th
[2]sns@kmutnb.ac.th

Abstract - **An automatic ontology matching is a task to align identical classes from different ontologies. This work presents a new method to detect same ontological class using components within ontology including label and property. Moreover, a synonym set gained from language resources such as WordNet and dictionary is included to detect synonymous classes with different surface form. These features are utilized to determine a similarity of classes in alignment. From testing, we found that the proposed method can automatically detect both explicitly and implicitly similar classes. The best combination of features is the exploitation of property and synonym for obtaining about 86% in f-measure while using only synonym gave highest precision for 92%.**

Keywords - **Ontology Matching, String Similarity, Property Comparison, Synonym Set**

I. INTRODUCTION

For decades, many ontologies have been developed and publically provided as domain-specific knowledge representation for sharing and system implementations [1]. Ontology [2] has become a famous knowledge base in several domains, such as biology, medicine, culture, and education. Some are large ontologies such as gene ontology [3], medicine ontology [4] and YAGO [5] while some are smaller but deeper in representing specific knowledge such as campus ontology [6], silk ontology [7], drug formulation ontology [8] and thalassemia ontology [9]. Mostly, these ontologies are used in expert systems, semantic webs, and knowledge based systems that are apparently demanded in several fields to include intelligence into services. Hence, the number of developed ontologies has regularly been increasing.

However, various ontologies were independently developed and specifically served to developers' purpose, but they could apparently contain similar content and scope. Ideally, reusing of existing ontologies is greatly suggested and expected since it will reduce time-consumption in a development cycle and provide a chance to improve existing knowledge to greater length. Unfortunately, the ontology developers often choose to develop their ontology from scratch since reading and editing existing ontologies is a great burden and a troublesome task due to different vocabulary usage and unrelatedly unfamiliar concepts in the existing ontology. Thus, there were researches on ontology mapping [10, 11] attempting to automatically find and link similar concepts in ontology.

The previous attempt [12] tried to exploit a list of synonyms for realizing the similar concepts in different ontologies despite using different labels. The work showed an impressive matching result, but it adds a heavy burden to linguists and ontologists to manually provide a synonym list covering all vocabularies used in ontology. In this work, we aim to overcome the issue by focusing on an automatic method to realize a similarity of the concepts without the use of human experts.

II. BACKGROUND

In this section, background of related technologies is mentioned. This includes brief details of ontology representation and existing works on ontology matching.

A. Ontology

Ontology schema [13] is a knowledge representation of related concepts in formal structure. To form a semantic relation of concepts, basic relations and components are constructed into a network of concepts representing in machine-readable class in schema.

In ontology, key components are class and relation [1, 2]. An ontological class represents a concepts distinguished from another while relation denotes a semantic link of one class to another. With relations of classes, a logical network of concepts based on fact is established to form a schema of knowledge in a domain. Since ontology is mainly about concepts in a domain, the representation of a concept is the label of a class; hence, using same label within ontology is not allowed [14].

Nowadays, ontology becomes a popular knowledge representation used in intelligent systems and used as information transfer. Since they are about concepts, it is highly possible that the classes in ontology maybe the same even though they belong to different domain. Since one of ontology strong points is that it can be fully reusable [2], there is a trend to encourage ontology engineers or users to use existing ontologies or at least to reuse some parts of ontology.

B. Ontology Matching

In ontology development, many classes are designed with specification of concepts in the interested domain and scope [14]. It is possible that ontologies of a similar domain may contain same or similar classes. Thus, to enlarge or combining ontologies, detection of the same or similar classes among different ontologies is greatly recommended.

There were several attempts on automatic matching same ontology classes in the past. The challenge in the task contains two main issues. The first one is to automatically find same classes with similar labeling and surface. The second one is to find a class with same semantic meaning of different class labels.

From existing researches, text similarity [10] was often applied to find the similar labels of ontology classes. With text similarity measurement, finding classes with similar labels can be done automatically. The commonly used text-similarity calculation is longest common subsequence (LCS) [15] since labels of ontology classes are composed of a string into word or short phrase. Text-similarity can also solve on an issue of finding minor typos or difference of language styles, i.e. British English and American English usage. However, it apparently cannot help on a totally different word usage.

For the second issue, one of the exisiting researches [12] used a manually created synonym set (SynSet) [16] of word to help matching classes with different surface label. A result of the work showed a potential in detecting similar concepts, but the accuracy of the result heavily relies on a quality of the SynSet. Since the work applied a list of SynSet by human experts, the result can cover as much as expertise of the experts as well as their burden. Another downside of the work is that there can be the case that the given ontologies may contain classes that are not well understood by SynSet makers, and they may miss important concepts that they are not familiar to.

III. METHODOLOGY

This work aims to automatically match ontology class from an ontology to another. By finding the similarity of the class, we focus on three aspects including label of the class, ontology structure and class composition. These aspects together can determine a similarity of classes from different ontologies.

A. Extracting Ontology Class

To compare two or more ontologies, we firstly need to read ontology classes and their relations. Ontologies are normally written in OWL format [17], upgraded version of XML according to W3C preference. OWL parsing is required to recognize ontology classes and relation based on class label. Because some ontologies may contain several languages for labels, we use the common language of the label for comparison.

In this work, we extract class labels using OWL parser [18] into a list as a main target for detecting class similarity. Each class includes the relations and properties. A list of each ontology will be compared to find a same for matching.

B. Label Similarity Calculation

This process is designed to find a class from different ontologies that may have a same or mostly similar label. This part will handle the explicitly similar class as one of the features in determining a class matching.

The similarity score is gained by calculating the commonly used strings of two classes. In this work, Levenshtein distance (LD) [19] is applied to give a string similarity score of two classes. LD is a measure of the similarity between two strings, referring as the source string (s) and the target string (t), respectively. The distance of two string is counted as the number of deletions, insertions, or substitutions required to transform (s) into (t). The algorithm of LD in stepwise is given in Fig. 1.

With LD, all classes from ontologies are compared, and we will obtain a string similarity score for each pair.

Step	Description
1	Set n to be the length of s. Set m to be the length of t. If n = 0, return m and exit. If m = 0, return n and exit. Construct a matrix containing 0..m rows and 0..n columns.
2	Initialize the first row to 0..n. Initialize the first column to 0..m.
3	Examine each character of s (i from 1 to n).
4	Examine each character of t (j from 1 to m).
5	If s[i] equals t[j], the cost is 0. If s[i] doesn't equal t[j], the cost is 1.
6	Set cell d[i,j] of the matrix equal to the minimum of: a. The cell immediately above plus 1: d[i-1,j] + 1. b. The cell immediately to the left plus 1: d[i,j-1] + 1. c. The cell diagonally above and to the left plus the cost: d[i-1,j-1] + cost.
7	After the iteration steps (3, 4, 5, 6) are complete, the distance is found in cell d[n,m].

Fig. 1 Levenshtein Distance Algorithm in Step-Wise [20]

C. Structure Similarity Calculation

In this process, we aim to find a similarity of class structure as another feature to detect class similarity. We expect that although classes may have different surface label, they are similar if they have same properties.

By comparing properties of classes, we gain percentage of common properties using property name (role-concept). We also include the properties inherited from their parent classes in comparison. We apply (1) to calculate the property similarity score.

$$PropertySim(X_i, Y_j) = \frac{common\ property\ (X_i, Y_j)}{total\ property\ (X_i, Y_j)} \quad (1)$$

D. Synonym Similarity Detection

This process aims to find the synonym classes with different labels and properties. In the existing work [12], the reference SynSet is assigned by human expert in which can be biased. In this work, we decide to use an existing bilingual dictionary and language resource in assisting on detecting synonym of the labels.

By applying WordNet [16], a set of synonym can be obtained directly. However, Thai WordNet [21, 22] though was mentioned in publications, but they are not completely finished or opened to use freely in public. Since this work involves in Thai labeled ontology comparison, we use bilingual dictionary to translate Thai label to English then we compare the translated labels with the SynSent gained from English WordNet. To increase a chance of matching, the translated to English label will limit to dictionary root form by removing English inflection such as tense and plurality form.

In case of assigning classes to the same concepts, both classes are apparently in the same synonym set or one level related for parent-child relation. Please see Fig. 2, for example of WordNet hierarchy relation.

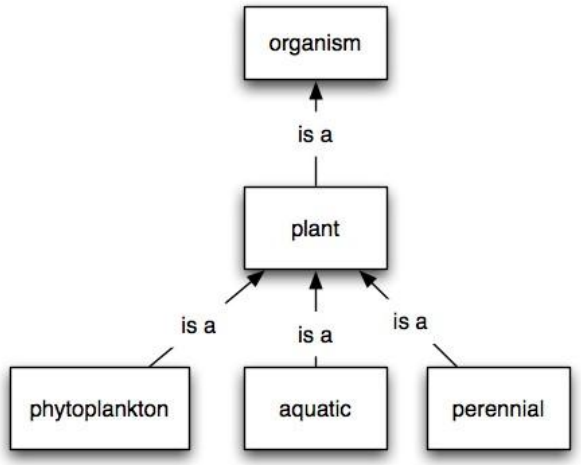

Fig. 2 WordNet hierarchy relation in which 'organism' is a one level parent node of 'plant' and 'aquatic' is a one level child node of 'plant'

An example of SynSet from English WordNet is shown in Table I. Please be noted that some set may be possible to contain either single label or several labels based on WordNet data.

TABLE I
EXAMPLES OF SYNSET AND ITS LABEL MEMBER GAINED FROM WORDNET

Set ID	Synonym List
1	'plant', 'flora', 'plant_life'
2	'dog', 'domestic dog', 'Canis familiaris'
3	'fruit'
4	'silk'
5	'cloth', 'fabric', 'textile'

E. Matching Similar Concepts

With three features mentioned above, we can determine classes from different ontologies for matching same class. For any class pair that passes one of the three criteria will be considered as same class.

The threshold for string similarity score for the same class is set as less than 0.15 while a percentage of the property similarity is assigned to 75%. For the case of SynSet matching, if labels are in the same set or a set within one level hierarchy relation (if the set is not directly taken by another pair), they will be considered for matching.

IV. EXPERIMENT AND DISCUSSION

To prove the usage of the proposed method, we selected two existing ontologies [7, 12] in a similar domain (silk product) for class matching. The details of these ontologies are given in Table II.

TABLE II
DETAILS OF ONTOLOGIES FOR EXPERIMENT

Ontology name	About	Class amount	Label Language
Silk Ontology [12]	Knowledge related to silkworm, silk product	78	Thai, English
Praewa Silk cloth [7]	Knowledge of silk cloth typology	47	Thai

From the datasets, we asked expert to manually match the classes as a gold standard to check with the matching result of our proposed method. The gold standard from human experts indicated that 29 class-pairs are same from those two ontologies. We thus gain

the f-measure score [ref] comparing to the gold standard by three features as StringSim (f1), PropertySim (f2), and Synonym (f3), respectively. The f-measure result is shown in Table III.

TABLE III
MATCHING RESULTS

	True Positive	False Positive	False Negative	Precision	Recall	F-measure
f1	13	9	16	0.59	0.45	0.51
f2	5	1	24	0.83	0.17	0.29
f3	22	2	7	0.92	0.76	0.83
f1-f2	17	10	12	0.63	0.59	0.61
f1-f3	22	9	7	0.71	0.76	0.73
f2-f3	24	3	5	0.89	0.83	0.86
All	24	10	5	0.71	0.83	0.76

From the result, we found that Synonym method shows the best result for single feature usage, and it gave highest precision, recall and f-measure. In combination of features, we found an interesting result that string similarity returned all correct matching pairs as a subset of correct results from synonym method. We hence plotted an image to represent the results of three features as shown in Fig. 3.

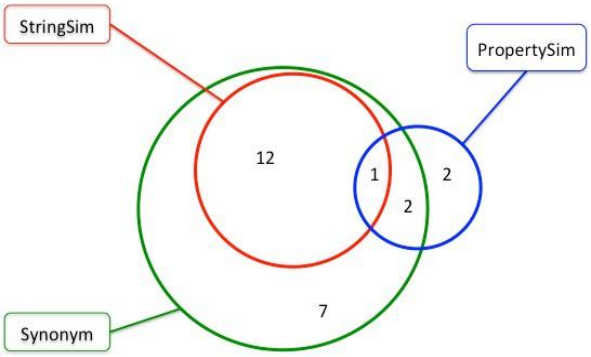

Fig. 3 Correct Matching Results Based on Features

We can see that the string similarity alone yielded adequate precision result, and it required lesser resource in calculation. it can increase overall result when combining with property similarity. However, it obviously got overrun by the synonym method in both single and combination result.

Moreover, the property similarity suffered from relatively low in recall score though its precision is noteworthy. In details, we found that both of the selected ontologies contain very few numbers of properties, especially Praewa Silk Cloth ontology. With insufficient amount of properties to be considered, the property similarity method cannot fully show its potential in matching classes. Despite that, this method is suggested to keep remaining in ontology class matching since it provides a good addition in obtaining a correct pair to other method with remarkable precision.

V. CONCLUSIONS

In this work, we present a method for automatic ontology matching. This method uses three features to determine if the class from two ontologies is the same. The features are label of the class, class property and meaning of class. Firstly, Levenshtein Distance is used for finding a similarity of class label considering commonness of explicit label string. Secondly, class properties are compared to represent structural similarity of class components. Lastly, a synonym set generated from WordNet is used as a reference to find ontology class with the same meaning despite having different surface label and property. With these three features, we conducted an experiment to see their potential in matching in both single and combination usage. The result showed that synonym method obtained the highest f-measure result, and combining property similarity and

synonym method gave the best result in terms of precision and recall.

REFERENCES

(Arranged in the order of citation in the same fashion as the case of Footnotes.)

[1] Gruber, T. (1993). "Towards principles for the design of ontologies used for knowledge sharing". In Guarino, N. and Poli, R. editors, Formal Ontology in Conceptual Analysis and Knowledge Representation. Kluwer Academic Publishers, Deventer, The Netherlands.

[2] Buranarach, M., Thein, Y., and Supnithi, T. (2013). "A Community-Driven Approach to Development of an Ontology-Based Application Management Framework". In: Takeda, H., Qu, Y., Mizoguchi, R., and Kitamura, Y. (eds.) Semantic Technology, Springer Berlin Heidelberg, pp. 306-312.

[3] Ashburner. and et al. (2000). "Gene ontology: tool for the unification of biology". Nat Genet 25(1): 25-9.

[4] Bodenreider, O. and Burgun, A. (2005). "Biomedical Ontologies". Springer-Verlag, pp. 211-236.

[5] Fabian, M., Suchanek., Kasneci, G., and Weikum, G. (2007). "Yago: A Core of Semantic Knowledge". In 16th international World Wide Web conference (WWW2007), New York, NY, USA, ACM Press.

[6] Somsuphaprungyos, S., Boonbrahm, S., and Ruangrajitpakorn, T. (2015). "An Ontology-based Frame-work of Intelligent Services for Smart Campus". In the Tenth International Conference on Knowledge, Information and Creativity Support Systems (KICSS2015).

[7] Singthongchai, J. and et al. (2012). "Development for Prae-Wa Silk Knowledge Base using Ontology".

[8] Chalortham, N., Leesawat, P., Ruangrajitpakorn, T., and Supnithi, T. (2011). "A Framework of Ontology-Based Tablet Production Supporting System for a Drug Reformulation". IEICE Transactions on information and Systems. Vol. E94-D, No. 3, pp. 448-455.

[9] Assawamakin, A., Chalortham, N., Ruangrajitpakorn, T., Limwongse, C., Supnithi, T., and Tongsima, S. (2011). "A development of knowledge representation for thalassemia prevention and control program". in Proc. 7th Int. Conf. on Natural Language Processing and Knowledge Engineering (NLP-KE), pp. 190-193.

[10] Euzenat, J. and Shvaiko, P. (2007). "Ontology matching". Springer-Verlag.

[11] Ji, Q., Haase, P., and Qi, G. (2008). "Combination of similarity measures in ontology matching using the OWA Operator". in Proceedings of the 12th International Conference on Information Processing and Management of Uncertainty in Knowledge-Base Systems (IPMU'08).

[12] Jaiboonlue, P. and et al. (2014). "A Framework of Automatic Alignment of Concept in Ontology with Confidence Score based on Inner Concept Information". Proceeding of JIST2014.

[13] Mizoguchi, K.T. (2003). "Tutorial on ontological engineering - Part 1: Introduction to Ontological Engineering". New Generation Computing, OhmSha&Springer, Vol. 21, No. 4, pp. 365-384.

[14] Noy, N. and McGuinness, D. "Ontology Development 101: A Guide to Creating Your First Ontology". Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report.

[15] Hirschberg, D.S. (1975). "A linear space algorithm for computing maximal common subsequences". Communications of the ACM. 18 (6): pp. 341-343.

[16] Miller, G.A. (1995). "WordNet: A Lexical Database for English". Communications of the ACM Vol. 38, No. 11, pp. 39-41.

[17] McGuinness, D. and Harmelen, F. "OWL Web Ontology Language

Overview". <https://www.w3.org/TR/2004/REC-owl- features-20040210/>.
[18] OWL API. <http://owlapi.sourceforge.net/>.
[19] Levenshtein, V.I. (1996). "Binary codes capable of correcting deletions, insertions, and reversals". Soviet Physics Doklady. 10 (8): pp. 707-710.
[20] Gilleland, M. "Levenshtein Distance, in Three Flavors". <http://people.cs.pitt.edu/~kirk/cs1501/Pruhs/Fall2006/Assignments/editdistance/Levenshtein%20Distance.htm>.
[21] AsianWordNet Project. <http://www.asianwordnet.org/>.
[22] Leenoi, D. and et al. (2008). "Building a Gold Standard for Thai WordNet". Proceeding of the International Conference on Asian Language Processing 2008 (IALP2008), pp. 78-82.