Triple Patterns Extraction for Accessing Data on Ontology

Zin Thu Thu Myint and Kay Khaing Win

***Abstract*—Some web applications use the ontology to integrate multiple data sources because ontology-based data integration is the ideal solution to handle the semantic conflict. For accessing the data on ontology, ontology understanding query such as SPARQL is needed. However, end users enter the unstructured sentence (words, statements, etc.) as an input when they wanted to search the required information on the web. So, it is needed to extract the triplets (i.e. subjects, predicates and objects) from the input query to build the ontology browsing query SPARQL. Although there are many triplet extraction algorithms, either they can't fully define all triple patterns from the incoming query or they are time consuming process. The proposed algorithm presented in this paper can handle this triplet's incompleteness problem and the aim of this system is to extract the specific triplets from incoming query and to add the necessary information for supporting SPARQL query generating process in a time-saving manner.**

***Index Terms*—Triplet extraction, ontology, data integration, semantic similarity, query translation.**

I. Introduction

Ontology can provide a common interface to retrieve the information located in separated data sources. That can also handle the semantic conflicts when integrating multiple data sources. So, many web applications use ontology as a standard tool for data integration [1]. Ontology can also retain the information about the concepts and relations of the multiple data sources. In order to access the data on ontology, ontology understanding query such as SPARQL is needed. However, End users cannot understand and type the format of SPARQL query explicitly when they wanted to search the required information. For this reason, many query translation procedures must arise in the query service of ontology-based data integration system. In query translation procedure, it becomes a majority role to extract the triple patterns from the input sentence. The goal of the system presented in this paper is to support time-saving triplet extraction algorithm for achieving sufficient triplets from the input query.

As architecture, this system follows the framework of Global As a View (GAV) approach of ontology. At each local source, it pre-builds the local ontology and the mapping schema that will be used to connect the global and local ontology, and to handle the semantic conflicts. At the cooperative source, it dynamically builds the global ontology by using the data received from the local site and supports the time-saving triplet extraction algorithm for extracting the triple patterns from the user input query. Then the mediator automatically generates the SPARQL query to browse the ontology by using the triplets obtained from the triplet extraction algorithm. After the SPARQL query has been generated, the data included in the ontology can be retrieved and returned to the user.

There are three main processing phases in ontology based data integration system such as ontology creation, ontology mapping and query service. Extracting the triplets from the user query is one of the most important roles of query service as describe in above. When the user query is submitted to the system, this query is needed to translate to the ontology understanding query SPARQL. The query language SPARQL has a graph-based structure and can be built by combining triple patterns extracted from the user input query [2]. If the user input query is unstructured query, it will be more difficult to define whether which words are subjects or objects. The proposed algorithm is able to identify the subjects, predicates and objects in the unstructured query exactly.

Currently there are many triplets extraction algorithm for supporting the triple patterns to assist the necessary application. They implement the triplet extraction process based on the parse tree generated by the parser. In this proposed algorithm, triple patterns included in the user input query are extracted with the help of domain specific ontology and machine readable dictionary WorldNet instead of using the parser. So, the processing time of this algorithm is lesser than other parser-based approaches.

This paper is organized as follows. In Section II, this paper presents the related work. The system design detail is presented in Section III. It gives in Section IV the description of proposed triplet extraction algorithm. Section V demonstrates the sample query testing and results of proposed algorithm by comparing the result of other competitive algorithms. Section VI concludes with some remarks.

II. Related Work

There are several approaches for ontology-based data integration system. The survey paper of Agustina… in [3] presents seven systems and three proposals of ontology based data integration systems. This paper describes the general framework of each system and makes comparison and classification for those systems. This paper can assist to understand the use of ontology and the processing phases that must be involved in the ontology-based data integration system.

It can see that query processing is one of the most important phases for accessing the data on integrated ontology. Here, it is needed to extract the triplets from the

Manuscript received December 31, 2012; revised February 23, 2013. This work was supported in part by the U.S. Department of Commerce under Grant BS123456.

The authors are with the University of Technology (Yadanarpon Cyber City), near Pyin Oo Lwin, Myanmar (e-mail: zinthumyint@gmail.com, kkhaingwin@gmail.com).

DOI: 10.7763/IJFCC.2014.V3.264

unstructured input query to generate the ontology understanding query. So, many approaches must be raised for extracting triplets from the input sentence. They are based on the parser and the processing time of these approaches are greater. The following relevant projects can be noted.

Delia... in [4] proposed triplets extraction algorithms based on the dependency tree generated by the Treebank parser [5] [6], Link parser and Minipar. Treebank parser generates the dependency tree for the sentence that has a syntactic structure. The complexity of the parsing time is $O(n^3)$ where n is the length of the input sentence [7], [8]. When applying the triplet extraction algorithm they proposed, it assumes the first noun in noun phrase (NP) subtree as subject, the deepest verb in verb phrase (VP) subtree as predicate and the first noun in propositional phrase (PP) or noun phrase (NP) subtree obtained in the VP subtree as object. This algorithm's time complexity is $O(n + (n-1))$, so worst case complexity is $O(n)$. Total time complexity is $O(n^3 + n)$; worst case complexity is $O(n^3)$. And this algorithm cannot fully extract all triplets from the query sentences because it cannot recognize if that input sentence may contain more than one subject, predicate and object.

They next algorithm they presented is based on the Minipar. Minipar takes the time complexity $O(n^3)$ to generate the parse tree [9] where n is the length of sentence; and the parse tree generated by Minipar contains the labels that represent the node, the word, the root and, grammatical category and relation. This algorithm defines the nodes with the grammatical relation '*s*' the subject element, the nodes with the relation '*i*' as the predicates and the nodes with the relation '*pcomp*' as the objects. Time complexity of this algorithm is $O((n-1)^2)$ where n is number of nodes, so worst case complexity is $O(n^2)$. So, the total time complexity becomes $O(n^3 + n^2)$; worst case is $O(n^3)$. This algorithm can extract all triplets from the query sentences.

Lorand Dali… in [10] presented the machine learning approach to extract triplet form sentence. The approach SVM is used to train a model on human annotated triplets and the features that are used in training data are computed from three parsers described in above. By applying this approach, the probabilities for every word which was built as triplet candidates are correct with high probability but an exhaustive search of all combinations would increases the execution time. Time complexity of SVM method is always $O(n^2)$ [11]. Time complexity of exhaustive search of all combination in this approach is $O(n^3)$ where n is the number words after removing the stop words from the original input query sentence. Total time complexity is $O(n^3 + n^2)$; worst case complexity is $O(n^3)$. This approach can extract all triplets from the query sentence.

To overcome this time consuming problems, it needs to avoid the use of parser to parse the input sentence and predefine the subject groups, object group and the set of predicates related to the corresponding subject or object by using the semantic ontology. The proposed algorithm will be presented in next section, extract the triplets from the input sentence with the help of domain specific ontology and can reduce the processing time by only detecting noun form of words from the input sentence with the help of WordNet instead of using parser.

III. Accessing Data on Integrated Ontology

Using ontology in data integration systems is an ideal solution to handle the semantic conflicts between various data sources. There are two trends to use the ontology in data integration system: one use for translating query or their result and the other uses ontology for the generation of global schema [12]. The system presented in this paper uses both of these two trends for data integration and accessing data on integrated ontology. The system architecture is depicted in Fig. 1.

As architecture, the system follows the framework of Global As a View (GAV) approach [13]. Local ontology is firstly created to represent the relational structure of database at each local source as the semantic model: table names are recognized as the ontology classes, column name in each table are recognized as the data-type properties of each class that are defined for the corresponding table and the between the classes are defined by the object-type properties. Global ontology is built by using the data collected from the existing local ontology [14]. Here, locally maintained ontologies are evolving independently, so this will need to reflect the global ontology.

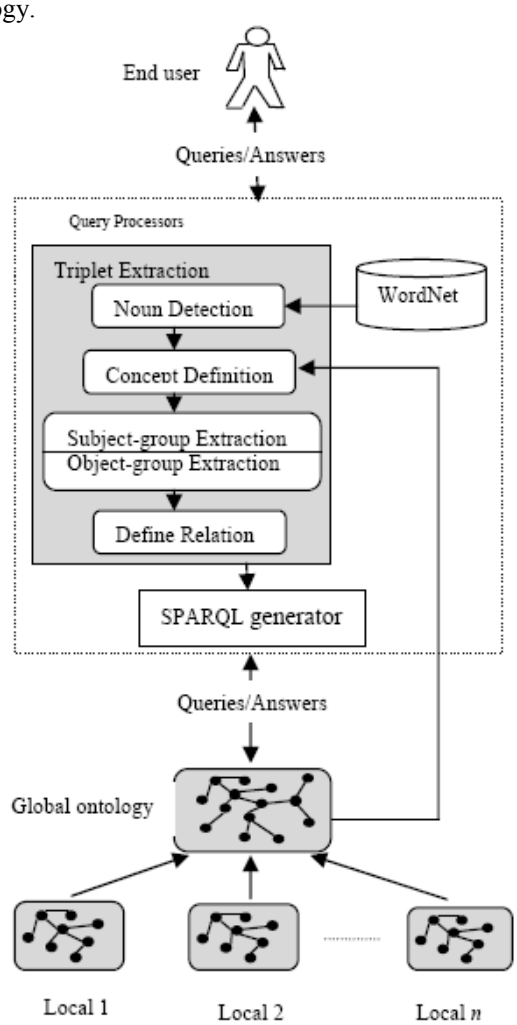

Fig. 1. Accessing data on integrated ontology.

For this reason, some activities would be took place in this system such as analysis of new contents of local ontology and identify what needs to be changed, added, modified or deleted to the global level.

When users make queries and submit them to the system, the global ontology and mapping schema are used to retrieve

the information needed from the sources. Mapping rules mean to construct equivalent, homonomy, overlapping, hyponomy and whole-part etc. between heterogeneous domain ontology concepts [15], [16]. These mapping rules are constructed by referring to the semantic similarity.

Moreover, users' input query may be an unstructured query (e.g. words, sentences, etc.). So, it is needed to translate the ontology understanding query such as SPARQL. SPARQL query language has a graph-based structure and can be built by combining triple patterns (subjects, predicates and objects). End user cannot write the SPARQL query directly because all of them cannot understand the structure of the SPARQL query. For this reason, query translation procedure is important for accessing data on ontology. According to the structure of SPARQL, it is firstly needed to extract the specific triplets from the user input query. This triplet extraction process will be fully described in the next section. After achieving the triplets from the input sentence, these are used to build the ontology understanding query SPARQL.

IV. Triple Pattern Extraction

As described in Section I, it is important to build the ontology for solving the semantic conflicts when separated data sources are integrated. For this reason, triple patterns extraction is needed to transform the incoming query to ontology browsing query (SPARQL). In this section, how triple patterns are extracted from the input sentence is explained and the triple pattern extraction algorithm is shown in Fig. 2. This proposed algorithm consists of three main parts: 1) concept identification, 2) subject group recognition and 3) object group recognition. It serves in the query web service when the user input query is submitted to the system.

A. Concept Definition

In this portion, each word in the input sentence is detected to clarify whether it has noun form with the help of machine readable dictionary WordNet. This processing step is shown at line 5 in algorithm. And then it defines the concept for each word of noun form by using the domain specific ontology. The concept of each word is defined as follows:

If the word extracted from the input sentence is similar to class type of concept in ontology, this word is defined as class.

If the word from the input sentence is similar to the data-type properties, this word is defined as predicates.

If the word from the input sentence is not appropriate with the type of class or data-type properties or object-type properties contained in the constitution of the domain specific ontology, this words is defined as constraint.

This concept definition step is shown at line 7 in algorithm.

Here, it uses the method Edit-distance and the dictionary WordNet to estimate the similarity between the words extracted from the input sentence and the concepts obtained in the ontology. The method Edit-distance is used to compute the similarity between the words which has no meaning (e.g. the name of the person) and containing spaces. Word sense similarity is guessed by checking the parent to subclasses relation and with the help of WordNet.

And it is also needed to estimate the similarity for each one word. The similarity between the two concepts is calculated by using the following equation supported by HowNet algorithm [17].

$$sim\,(s_1, s_2) = \sum_{i=1}^{4} \beta_i \prod_{j=1}^{i} sim_j\,(s_1, s_2) \qquad (1)$$

where $\beta_i\,(1 \le i \le 4)$ is an adjustable parameter; moreover, $\beta_1 + \beta_2 + \beta_3 + \beta_4 = 1$, $\beta_1 \ge \beta_2 \ge \beta_3 \ge \beta_4$, which reflects the degree contributions to the overall semantic similarity in descending order from sim_1 to sim_2. And $sim_j(s_1, s_2)$ is respectively semantic similarity of four parts of original concept atoms [18]. The threshold value for semantic similarity of two concepts is defined as 0.5.

```
1.  Algorithm: Triple-Pattern-Extraction. Generate subject, predicate, object and value
    of unstructured query.

2.  Input:
        Query Sentence
3.  Output:
        Subject, predicate, object and value

4.  Function: Triple Extraction (Query Sentence)
5.  Noun Detection.          <by WordNet>
6.  Create List for each word.
7.  ConceptList<-DefineType-of-Concept(List)  <by ontology>
8.  Object <- Extract-OV (ConceptList)
9.  Subject-group <- List
10. Subject <- EXTRACT-SP (Subject-group)
11. result <- result U Subject U Object
12. Return result

13. Function: EXTRACT-SP (Subject-group)
14.   while (predicate.exist)
15.  Find the class of predicates                    <by ontology>
16.  cluster the predicates with the same class
17.  Subject <- class name
18.        If Subject is equal to class-type (word) in the list
19.      Remove this class-type(word) from the list
20.  Predicate <- cluster of predicates
21.  Remove predicates from the list
22.  Define the relation based on range value   <by ontology>
23.  Result <- Result U Subject U Predicate U relation
24.   while (class.exist)
25.  Subject <- class name
26.  Remove this class-type(word) from the list
27.  Predicate <- string type of data-type properties
28.  Define the relation based on range value   <by ontology>
29.  Result <- Result U Subject U Predicate U relation
30. return result.

31. Function: Extract-OV(ConceptList)
32.   While( constraint. exist)
33.      If consecutive constraints are found then
34.           Decide on these constraints should be combine or not.

35.       Object <- constraint.class                    <by ontology>
36.       Define relation based on range value          <by ontology>
37.       Predicate <- constraint.datatype-properties   <by ontology>

38.      If Object and Predicate are equal to class-type (word) in the list
39.        Remove resulted object and predicates from the list
40.      Result <- Result U Object U Predicate U constraint U relation
41.      Remove constraint from List

42. return Result
```

Fig. 2. The algorithm for extracting triplets from unstructured input query.

B. Object Group Recognition

In this portion, the function of extracting triplets from the input sentence operates for recognizing the corresponding objects, the relations to these objects and predicates by connecting their constraints which are obtained from the concepts list defined in previous section. Object group extraction function is described detail at line 8 and lines 32 to 45 in algorithm.

Here, if consecutive constraints are found, these are determined to be combined or not by comparing the values in

the ontology. This condition is checked at lines 33 and 34 in algorithm.

Then the class and data-type properties in ontology correspond to that value are considered as the object and predicates of that value. The assignment statements of these values are shown at lines 35 and 37 in algorithm. The relation to this object is defined by ontology based on the range value in ontology at line 36 in algorithm.

Then it is needed to remove the facts related to the extracted objects, from the concept list. The decision to remove the corresponding values is made at lines 38 and 39 in algorithm. Finally the result of object groups is returned and this constraint type of word is removed from the concept list at lines 40, 41 and 42 in Algorithm. The remaining concept list is used to continuously extract the subject groups.

C. Subject Group Recognition

In this portion, the subjects and predicates required can be easily extracted. In algorithm, lines 9, 10 and 13 to 30 fully described about this function.

Here, two situations are considered: 1) containing all predicate values in the statement and 2) containing only the class without predicates.

In the first situation, it finds the class name corresponding predicates by using ontology in line 15 in algorithm. Then it groups the predicates which have the same class name in line 16 in algorithm. It assumes the class name as subject and resulted predicates group as relevant predicates of that class. If the subject and predicates are included in the list, then they are removed from the concept list. Finally, it defines the relation to that class (subject) based on range value in ontology and all of the result are union returned. These processing steps are shown at lines 17 to 23 in algorithm.

In second situation, if the class types of words are still remaining in the concept list, these classes are assumed as subjects and removed from the concept list. Then it chooses the string type of predicates corresponding to those classes in ontology as the relevant predicate for those classes. Finally, it defines the relation to that class (subject) based on range value in ontology and all of the result are union returned. These processing steps are shown at lines 24 to 30 in algorithm.

All triplet patterns (subjects, predicates and objects) extracted from the input sentence are used to build the SPARQL query for browsing the ontology. This proposed algorithm successfully extracts all triplets from the unstructured query.

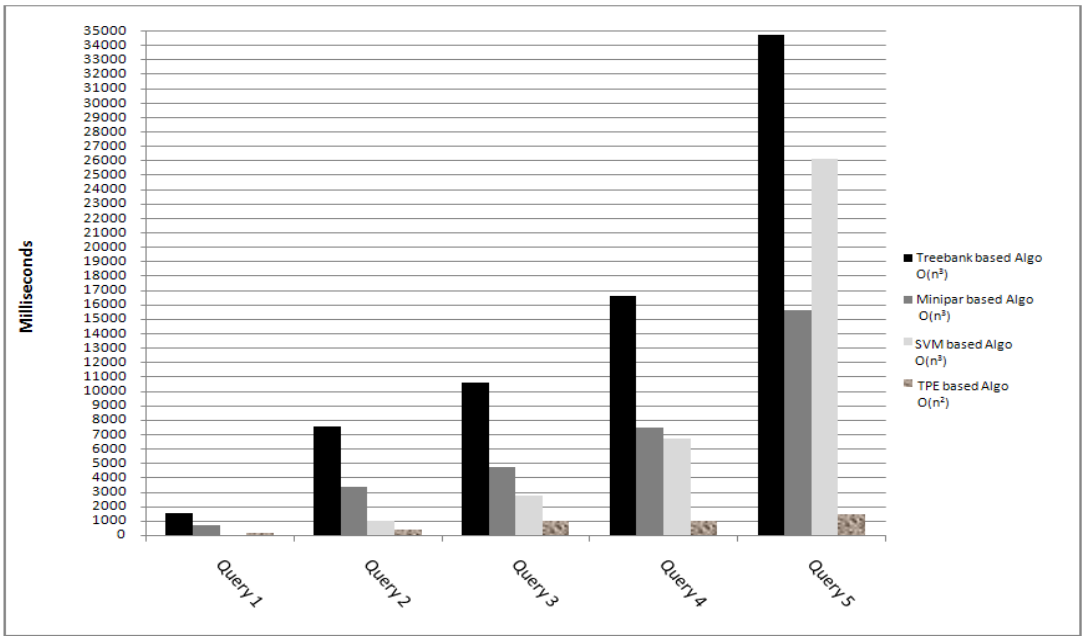

Fig. 3. Different query testing on triplet extraction algorithms that are based on treebank parser, minipar, SVM based approach and proposed triplet extraction algorithm TPE.

D. Performance Analysis

It is interesting to see the performance of proposed triplet extraction algorithm to other triplet extraction algorithms. The performance is measured on the processing time of extracting triplets from the input sentence. Firstly, the symbol 'n' is assumed as the number of words in the query sentence. In this algorithm, time complexity of noun detection from the input sentence is 'n' times. The step of concept definition takes 'n' times for the length of list that store the noun form of words obtained from the noun detection step. Here, it takes $n + (n-1)$ times for ontology graph traversal. So, concept definition step takes n^2 times totally. For object group recognition step, it also takes 'n' times as it is needed to search the concept list defined by concept definition step and it takes $n + (n-1)$ times for ontology graph traversal. So, object group recognition step takes n^2 times totally. Subject group recognition step takes 'n' times for searching in the remaining concept list and it takes $n + (n-1)$ times for ontology graph traversal. So, subject group recognition step also takes n^2 times totally. So, total time complexity for this algorithm is $O(n + 3n^2)$, worst case is $O(n^2)$.

Compared with other approaches described in related work: triplet extraction algorithm based on dependency tree generated by the Treebank parser with total time complexity $O(n^3)$, the algorithm based on the Minipar with total time complexity $O(n^3)$ and triplet extraction using SVM with total time complexity $O(n^3)$, the proposed triplet extraction algorithm is more time-saving than other algorithms.

V. Sample Query Testing

The analysis is made compared with the algorithms which are based on Treebank parser, Minipar and the approach of triplet extraction using SVM. Time complexity of each approach is computed on different types of unstructured query and the analytical result obtained by comparing 5 different unstructured queries are depicted in bar chat. These 5 different unstructured queries are:

Query1: All about John. ($n = 3$)

Query2: staff name at the software engineering department. ($n = 7$)

Query3: Company's names that are included in advance science and technology department. ($n = 11$)

Query4: name, age, NRC, father-name of staff who gets M. C. Sc degree and international paper acceptance. ($n = 14$)

Query5: Staff name, degree, position, start-year, department, compensation and bond-year who get English exam marks >50, Major exam marks >50 and had got Ph.D degree. ($n = 26$)

Treebank parser, Minipar and proposed triplet extraction algorithm count 'n' value as the number of all of words that are included in the input query statement. However, SVM-based approach counts 'n' value after removing the stop words from the input statement. So, 'n' values of SVM-based approach are 1, 5, 7, 11, and 22 respectively for queries 1 through 5. When these queries are tested by applying the triplet extraction algorithms that are previously described, the time taken to extract the triplet patterns for each algorithm is shown in figure 3.

By perceiving the results shown in figure 3, it can be see that all other triplet extraction algorithm takes more execution time than the proposed triplet extraction algorithm presented as 'TPE based Algo' in figure.

VI. Conclusion

This paper has presented the triplet extraction algorithm that can fully extract all triplets included in the unstructured sentence with the help of domain specific ontology and machine readable dictionary WordNet. Moreover, it makes comparison with other triplet extraction algorithms based on execution time by applying different types of query. This algorithm can be encapsulated in the web query services of ontology-based data integration system. By seeing the appraisal described in sample query testing, this algorithm takes the less execution time than the other algorithms and will be a useful algorithm for extracting triplets from the unstructured sentence.

Acknowledgment

I am very grateful to Dr. Aung Win and Dr. Soe Soe Khine for fruitful discussions during the preparation of this paper and also specially thank to Rector, Professors and colleagues from Technology University (Yadanarpon Cyber City), Myanmar.

References

[1] M. Gagnon, *Ontology-Based Integration of Data Sources*, 2007.
[2] M. E. Saleh, *Semantic-Based Query in Relational Database Using Ontology*, 2011.
[3] A. Buccella, A. Cechich, and N. R. Brisaboa, *Ontology-based Data Integration Methods: A Framework for Comparison*, 2003.
[4] D. Rusu, L. Dali, B. Fortuna, M. Grobelink, and D. Mladenic, *Triplet Extraction from Sentences*, 2008.
[5] Stanford Parser web page. [Online]. Available: http://nlp.stanford.edu/software/lex-parser.shtml.
[6] Sharp NLP. [Online]. Available: http;//www.codeplex.com/sharpnlp.
[7] M. Galley and C. D. Manning, "Quadratic-Time Dependency Parsing for Machine Translation," in *Proc. of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP*, pp. 773–781, Singapore, 2-7 August 2009.
[8] M. A. Covington, "A Fundamental Algorithm for Dependency Parsing," in *Proc. of the 39th Annual ACM Southeast Conference*, 2001.
[9] Books. [Online]. Available: http://books.google.com/books?id=X4xog4y8aIkC&pg=PA201&lpg=PA201&dq=Time+complexity+of+Minipar+parser.
[10] L. Dali and B. Fortuna, *Triplet Extraction from Sentence Using SVM*, 2008.
[11] D. Cao and D. Boley, *On Approximate Solutions to Support Vector Machines*, 2006.
[12] F. Hakimpour and A. Geppert, "Relsoving Semantic Heterogenity in Schema Integration: an Ontology Based Approach," in *Proc. of Conference on Formal Ontology in Information Systems*, Ogunquit, Maine, USA, October 17-19, 2001.
[13] P. Haase and B. Motik, "A Mapping System for the Integration of OWL-DL Ontologies," IHIS'05, Bremen, Germany, November 4, 2005.
[14] M. Uschold, *Creating, Integrating and Maintaining Local and Global Ontologies*, 2002.
[15] H. Wache, T. Vogele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Hubener, "Ontology-based data integration – A Survey of Existing Approaches," in *Proc. of IJCIA-01 Workshop: Ontologies and Information Sharing*, Seattle, WA, 2001.
[16] J. Lu, Y. Zhang, Z. Miao, and P. Zhou, *The Semantic Web Principle and Technology*, Science Press: Beijing, 2007.
[17] Y. Jiang and Y. Ding, "An Improved Computation Method of Word Semantic Similarity Based on How Net," *Journal of Chongqing University of Posts and Telecommunication*, 2009.
[18] J. Cai, Y. Zhang, F. Hu, and J. Dong, "Domain Ontology Mapping Based Semantic Web Mining Models Research," in *Proc. of International Conference on E-Business Intelligent*, China, 2010.

Zin Thu Thu Myint received Bachelor of Computer Science from University of Computer Studies; Yangon in Myanmar .She completed the Master Course from University of Computer Studies since 2008 and especially studied and finished the thesis by Machine Learning in Data Mining. She is working now in University of Computer Studies, Maubin as a tutor under Software Department. Now, she is now a Ph.D student in University of Technology (Yadanarpon Cyber City) near Pyin Oo Lwin , Upper Myanmar and mainly study about Software Engineering. Her fields of interest are Ontology, Web Mining, Data Mining and Query Processing.

Kay Khaing Win received Master of Computer Science from University of Computer Studies; Mandalay, Myanmar in 2002. She completed the Ph.D program from University of Computer Studies since 2007 and especially studied and finished the research by Special Semantic Web. She is working now in University of Technology, Yadanarpon Cyber City, near Pyin Oo Lwin, Upper Myanmar as a lecturer under Software Department. Her fields of interest are Ontology, Web Mining, Data Mining and Query Processing.