Journal of Computer Science and Cybernetics, V.34, N.3 (2018), 187–198
DOI 10.15625/1813-9663/34/3/13100

AN IMPROVEMENT METHOD FOR SEMANTIC MAPPING DATABASE TO ONTOLOGY*

PHAM THI THU THUY

Information Technology Faculty, Nha Trang University, Vietnam
thuthuy@ntu.edu.vn

Abstract. Enormous amount of available data in relational database (RDB) format creates a demand for automatic transforming them into Web Ontology Language (OWL) ontology to reuse in the Semantic Web. Many approaches have been proposed, however, most of them simply generate output ontology as the same flat structure with the original database and result in redundancy of ontology data. As an attempt to resolve the redundant problem, we propose a novel approach to generate OWL ontology from relational database while focusing on the similarity measure of duplicate attributes in relational tables. Experimental results show that the proposed method reliably predicts semantic similarity of duplicate columns and produces a better-quality OWL ontology.

Keywords. Relational database; OWL ontology; Mapping; Similarity measure.

1. INTRODUCTION

As reported in [1], 70% of current Web were backed up by relational database. Making this huge amount of data hosted in RDBs available to the Semantic Web has been an interesting field of study during the last decade. In this regards, OWL is a powerful pivot format. OWL data can be understood by the computer and then can be shared, exchanged or integrated into a data repository enabling applications to use the data in different contexts [2].

OWL facilitates greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDFS) by providing additional vocabulary along with a formal semantics. As in RDFS, the basic elements of OWL are classes, properties and individuals, which are members of classes. OWL properties are binary relationships and are distinguished in object and data type properties. Object properties relate two individuals, while data type properties relate an individual with a literal value [3]. Therefore, this paper uses OWL dialect to store the result.

Bridging the conceptual gap between the relational model and OWL play an important role in utilizing the existing data on the Semantic Web [4, 5]. There are some proposals that move relational data to the OWL dataset. The typical approaches are proposed by S. Zhou et al. [6], L. Lili et al. [7], El Idrissi B. et al. [8], M. Laclavik [9], DM-2-OWL [10],

*This paper is selected from the reports presented at the 11th National Conference on Fundamental and Applied Information Technology Research (FAIR'11), Thang Long University, 09 - 10/08/2018.

L. Zhang et al. [11]. However, most of those approaches are simple and equivalent translating: each table maps to a class, each row maps to an instance, and especially each column maps to an OWL data type property without considering that there are some relational columns have the same name and similarity with others. The backside of this simplicity may lead to data redundancy because duplicate columns may represent the same information. The perfect OWL generation should create a correct, complete, and unique representation of every concept. To obtain this data quality, a similarity computation of duplicate columns is used. In this computation, if two elements have highly similar semantics, they are transformed into one representation.

This paper proposes novel metrics to measure the similarity between duplicate columns in a RDB and presents a transforming strategy for each similarity level. Compare with the previous studies on generating OWL ontology from RDB, our method is a new technique that solves the duplicate problem efficiently and reliably. Furthermore, the proposed method produces a syntactically legal OWL ontology, which is easily processed and interpreted by semantic applications.

The rest of the paper is organized as follows. Section 2 presents some specific methods of the related work. Section 3 describes the details of OWL generation method, including the semantic similarity measurement for duplicate columns and the transformation of RDB into the OWL ontology. Section 4 presents the experimental setup and results. Finally, Section 5 concludes the paper.

2. RELATED WORK

There are many approaches investigating the transformation of the relational database into an OWL dataset. J. Sequeda et al. [12] propose a semi-automatic approach to translate relational database to RDF by using OWL vocabulary. This approach results in new OWL syntax and can translate some of relational tables. S. Zhou et al. [6] replies on Jena and some proposed rules to generate an OWL ontology from relational database. This proposal is direct mapping hence it is not semantic reserving because of losing the connection between primary key and foreign key. L. Lili et al. [7] and El Idrissi B. et al. [8] translate relational schemas to OWL ontology while maintaining the relationship between foreign key and primary key, but they does not transform the relational instances. M. Laclavik [9] uses SQL query to directly extract some relational data and store in RDF/OWL format. DM-2-OWL [10] is an automatic approach that proposes some rules to translate relational schema to OWL ontology. However, this method does not transform relational instances into OWL individuals. Other approaches [11, 13, 14, 15, 16, 17, 20, 21] also define some rules to translate relational tables into OWL ontology.

In general, most of the related approaches are semi-automatic approach by defining some rules to extract some relational schemas/instances to store in OWL ontology. Some approaches do not discover inheritance, thus generating an ontology that has the same flat structure as the original RDB. Other approaches can translate all relational tables but they are direct mapping which does not consider the connection between foreign key and primary key. Therefore, the results does not maintain the relationship between relations, or not semantic reserving.

In this paper, we attempt to resolve these problems by proposing a novel approach based

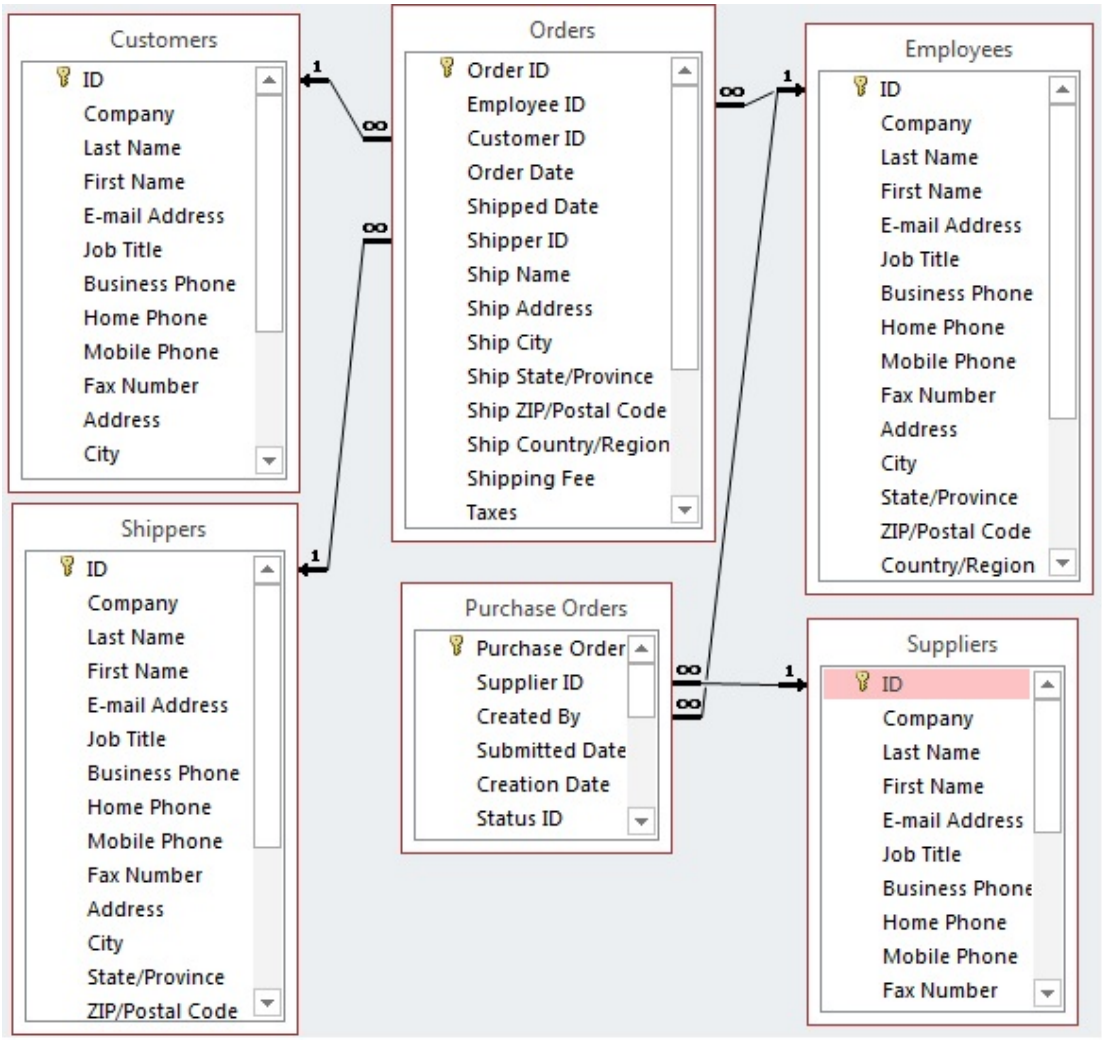

Figure 1. A part from the NorthWind sample database supported by Microsoft

on computing the similarity between duplicate columns that can extract some or all required information from relational database into OWL ontology without any user intervention. Moreover, our method maintains the relationship between foreign and primary keys among relations and uses XML format as the template to store relational instances then translate XML instances into OWL individuals.

3. METHOD FOR GENERATING OWL FROM RDB

3.1. Measuring the similarity of duplicate column

A relational database is a demonstration of the relational model. This model composes constructs for specifying tables, columns, data types, constraints and other semantics. In a RDB, a table may contain many columns. The column name in one table can be appeared in the other tables, as Figure 1 shows.

An ontology is a demonstration of an ontology model. This model includes classes, properties, data types, inheritance restrictions, and other semantics. According to most related approaches, each column in RDB (which is neither primary key nor foreign key) is transformed into a data-type property with same name corresponding to the column. The OWL vocabulary support each column name with a unique identifier (rdf:ID). However, this solution may lead to the data redundancy since the duplicate columns may express the same content.

For example, as presented in Fig. 1, the column Company, Last Name, First Name, E-mail Address, Job Title, Business Phone, Home Phone, Mobile Phone, Fax Number, Address, City,... of the table Customers are similar to those column names in tables Employees, Suppliers, and Shippers. These columns have the same names but may be different in table names and data types. Moreover, although the names of those tables are different, they are quite similar in the semantics because they describe about human being.

On the basis of the above mentioned observations, we can conclude that there are two main factors that affect the similarity between duplicate columns: the table name and the data types. Therefore, our duplicate column similarity measure is the combination of these two factors using a weighted function, which is determined by Definition 1.

Definition 1. The duplicate column similarity (ColSim) between duplicate columns, $C1$ and $C2$, between tables, $T1$ and $T2$, in a RDB is defined as the weighted sum of their table name similarity ($TaSim$) and their data-types similarity ($DaSim$)

$$ColSim(C_1, C_2) = \alpha \times TaSim(C_1, C_2) + (1 - \alpha) \times DaSim(C_1, C_2) \tag{1}$$

where α is the weight factor. If the *TaSim* property contributes more than the *DaSim* property to the similarity of duplicate columns, then the weight α of the *TaSim* is greater than the *DaSim*s weight. Without loss of generality, in our implementation, we assume that all similarity properties have an equivalent role; thus, the weights α =0.5.

To compute the name similarity between two tables, $T1$ and $T2$ of the corresponding two columns, $C1$ and $C2$, we measure the meaning of the table names by referring them in the WordNet [18]. We reuse the distance-based method [19] to measure the distance similarity of table names in the WordNet taxonomy. The name similarity between two tables, $T1$ and $T2$ of the corresponding two columns, $C1$ and $C2$, is determined by the following

$$TaSim(C_1, C_2) = \frac{2 \times depth(LCS)}{depth(C_1) + depth(C_2)} \tag{2}$$

where *depth(LCS)* is the number of nodes from the common super-concept of element $C1$ and $C2$ to the root node; $depth(C1)$ and $depth(C2)$ are numbers of nodes from element $C1$ and $C2$ to the root node.

In some cases, the table name is a combination of words or the short form of some words, so the normalization steps are required. These steps remove genitives, punctuation, capitalization, stop words and inflection (plurals and verb conjugations), and replace the short word by its full name.

The second factor that affects the similarity between two columns is the data-types similarity ($DaSim$). Since in a RDB, if two columns have different data types, they cannot make the reference to each other. Therefore, we assume that if the data types of two columns, $C1$ and $C2$, are different, their $DaSim(C1, C2) = 0$, otherwise $DaSim(C1, C2) = 1$.

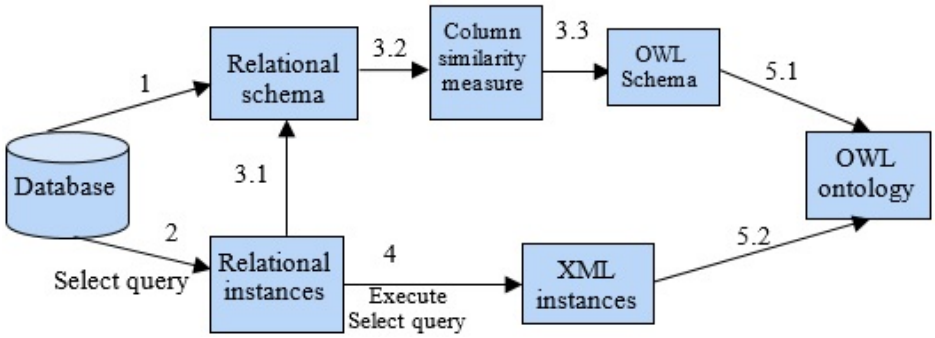

Figure 2. The framework of OWL generation from relational database

Depending on the expected similarity value, the duplicate columns can be classified into two groups, similar and non-similar. The generation strategies are then applied to transform these duplicates into the appropriate OWL concepts. In this paper, we use the threshold value 0.6 to classify the duplicate columns. If two duplicate columns have the similarity score above 0.6, they are similar, otherwise they are non-similar. We choose the value of 0.6 because in our proposed measure, if two duplicate columns have the same data types, their DaSim value is already 0.5, so if their table names are little similar, they could be considered as similar to each other.

3.2. RDB to OWL framework

The details of our approach is presented in Fig. 2.
As shown in Fig. 2, our approach has five small steps as follows:

- Step 1: Describe all attributes in a database. The description result is a text file stored in secondary memory.
- Step 2: Use SELECT command to extract the data describing the resource. Each resource should belong to a primary table. The attributes of the primary table must be extracted first.
- Step 3: Compute the column similarity and generate OWL Schema (OWLS) file based on the description file (Step 1) and the attributes extracted from Step 2.
- Step 4: Execute the SELECT query to extract instances in the relational database. The results are stored in XML format.
- Step 5: Generate OWL dataset from OWLS and XML files.

Details of each step are presented in next steps.

SELECT syntax for extracting data

The SELECT command must contain the primary key of the primary table. This primary key is considered as URI for instances in the resource. The SELECT syntax is as follows:

SELECT tableName.ID, attribute1 As AliasName1, attribute2 As AliasName2....
FROM tableName, table2Name ....

WHERE .....
FOR XML AUTO, ELEMENTS ODER BY tableNameID

We note that the attribute key in the primary table must be extracted in the SELECT command.

Generating OWLS

We assume that the extraction of data in the relational database is not redundant. It means that if the foreign key is extracted, the primary key which is referenced by the foreign key is not extracted and vice versa. The OWLS file contains the classes and properties which are described as follows:

- Description of classes: Each table in a database is transformed into a class. The description of a class is based on the key attribute (primary key or foreign key). The class name is a value in the range column. If the parent contains values, the class in the range column is a child of a parent class.

- Description of properties: The domain of all the attributes is the name of primary table. The range of attributes is the values from the range column.

Generating OWL

This step produces an OWL file from the XML and OWLS files generated in the previous step. The SELECT command to generate OWL format in the form of XML file is as follows:

SELECT property1, property2.....
FROM table1, table2, ...
WHERE [Where conditions]
ORDER BY property1
FOR XML AUTO, ELEMENTS

where *property1* is the attribute key in the primary table; *table1* is the name of primary table. The algorithm to generate OWL file can be described as the following pseudo codes:

```
Algorithm 1. GenerateOWL
 Input: An XML file Fxml, a primary table Tp, a OWLS file Fs
 Output: An OWL file F
1: Collect all the children elements of the root element in the XML file Fxml.
2: FOR each child element ec in Fxml:
3:      Read  the value ID of the attribute key in the Tp;
4:      Create a resource having URI=baseURI+ResourceName+#+ID;
5:      FOR each property p in Fs:
6:           Take a list of elements (listEle) in an instance;
7:           IF n(|listEle|) > 0 THEN
8:                FOR each child element ec in listEle:
9:                     IF p is the attribute key THEN
10:                         Create a corresponding property;
11:                         Generate a property having ecs value.
12:                         Append predicate to the resource.
10. RETURN  F
```

The Algorithm 1 describes the steps to generate an OWL file from the primary table, the OWLS file and the XML file. First, all the children elements of the root element in XML file (line 1) are collected. Second, the value ID of the attribute key in the primary table is read, and then a resource having an URI that includes base URI and Resource Name and ID for each child element in the XML file (line 2-4) are created. Third, a list of elements (listEle) in an instance for each property (line 6) is taken. If the number of elements in listEle is greater than 0 and the property is the attribute value, a corresponding property which referenced to the resource having URI for each property (line 7-10) is created. The property whose value is the value of element child in listEle is generated, and then predicate between containers (line 11) is also created. Finally, all predicates are appended to the resource. If all the properties in OWLS are not traveled, the algorithm will return to the attribute key checking step (line 12). Through all these steps, the OWL file result is generated.

3.3. Converting data type

The transformation of RDB into OWL ontology requires to reserve the information about data types. In this study, our RDB is implemented on SQL database management system, so we use the data types supported by the SQL to express the data types for each column. Moreover, since OWL does not have the defined data types, OWL uses the data types of the XML Schema (XSD). Table 1 presents some common data types in SQL corresponding to the data types in XSD.

Table 1. Mapping data types from SQL to XSD

SQL data type	XSD data type
Number	
Decimal, Numeric	Decimal
Real	Float
Float	Double
Integer, Int	Integer, positiveInteger, negativeInteger
BigInt	Long
SmallInt	Short
TinyInt	UnsignedByte
Character, string	
Char, VarChar	String
Nchar, nvarchar, text, ntext	String
Date, Time	
DateTime	DateTime
Date	Data
Time	Time
Other data types	
Binary, VarBinary	Base64Binary
Bit	Boolean
Variant	anyType
Interval	Duration

4. EVALUATION

The proposed transforming method is evaluated by matching a relational database with an OWL file to determine the true matches, and compare the results with the related methods. To assess the quality of the matching system, the *precision* and *recall*[1] are used. Given the set of expected matching pairs, R, (produced by a human), the set of alignment pairs, T, (produced by the matching system for the proposed methods), the *precision* is computed as the following equation:

$$precision(R,T) = \frac{|R \cap T|}{|T|}. \tag{3}$$

Recall specifies the share of real correspondences

$$recall(R,T) = \frac{|R \cap T|}{|R|}. \tag{4}$$

Although *precision* and *recall* are the most widely used measures, when comparing matching systems, one may prefer to have only a single measure. Moreover, systems are not comparable based solely on *precision* and *recall*. For this reason, *F-measure* is introduced to aggregate *precision* and *recall*. *F-measure* presents the harmonic mean of *precision* and *recall*

$$F - measure = 2 \times \frac{precision \times recall}{precision + recall}. \tag{5}$$

To obtain practical evidence, we applied our transformation to two sample databases produced by Microsoft, particularly, *Northwind*[2], and *Pubs*[3].

We compare the *precision*, *recall*, and *F-measure* values between our proposed method and the most related work, such as S. Zhou et al. [6], L. Lili et al. [7], M. Laclavik [9], and DM-2-OWL [10]. The matching system is also implemented by using Visual C#. The comparing results are shown in the following figures, Fig. 3 and Fig. 4.

Fig. 3 and Fig. 4 show that our matching quality is highest in comparing to those of the related work. L. Lili et al. [7] is ranked second, then S. Zhou et al. [6], DM-2-OWL [10], and M. Laclavik [9]. The main reason is that our method and S. Zhou et al. [6] transform all relational database into OWL whereas the approaches [7, 10] only translate the relational schemas to OWL dialect, the left approach [9] extracts some relational tuples. Moreover, our method maintains the relationships between foreign key and primary key among relations whereas the approaches [6, 9], and [10] do not. Among S. Zhou et al.[6], L. Lili et al. [7], and DM-2-OWL [10] methods, L. Lili et al. [7] gives the highest matching values since this method retains the connections between foreign keys and primary keys. Moreover, when extracting some portions of the relational data, those three methods change some of the data structure so that their matching scores are not good.

There are some small differences between Fig. 3 and Fig. 4, since the differences of Northwind and Pubs databases. Northwind database has 13 relations in comparing to 11 relations in Pubs database. Among those relations, there are relationships between foreign

[1]http://en.wikipedia.org/wiki/Precision_and_recall

[2]http://northwinddatabase.codeplex.com/

[3]http://technet.microsoft.com/en-us/library/aa238305%28v=sql.80%29.aspx

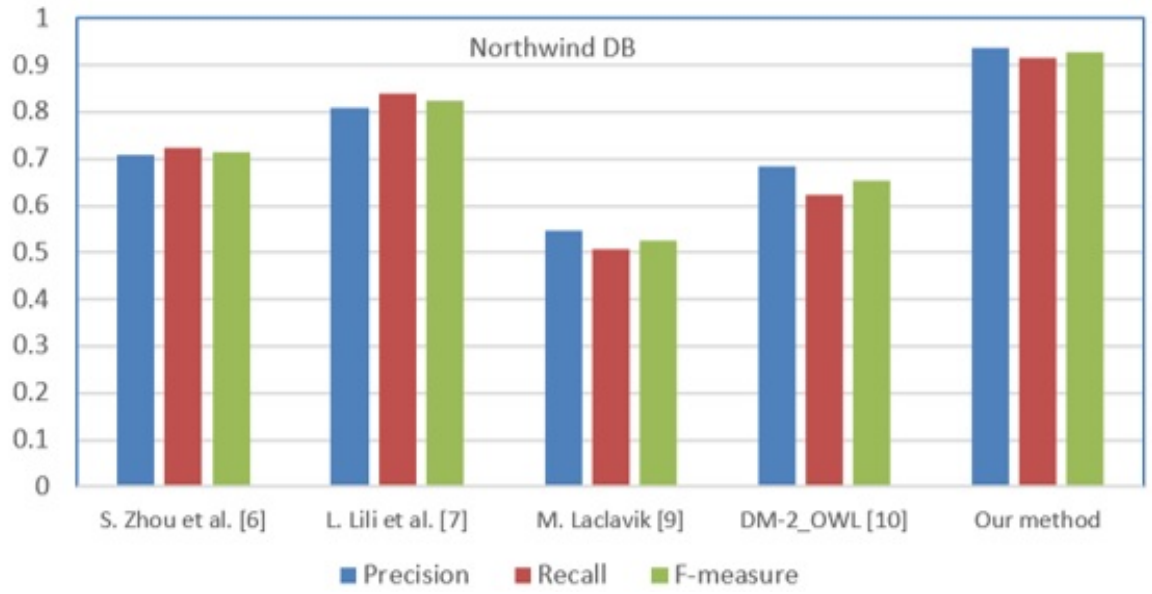

Figure 3. Matching comparison between our method and related work on Northwind database

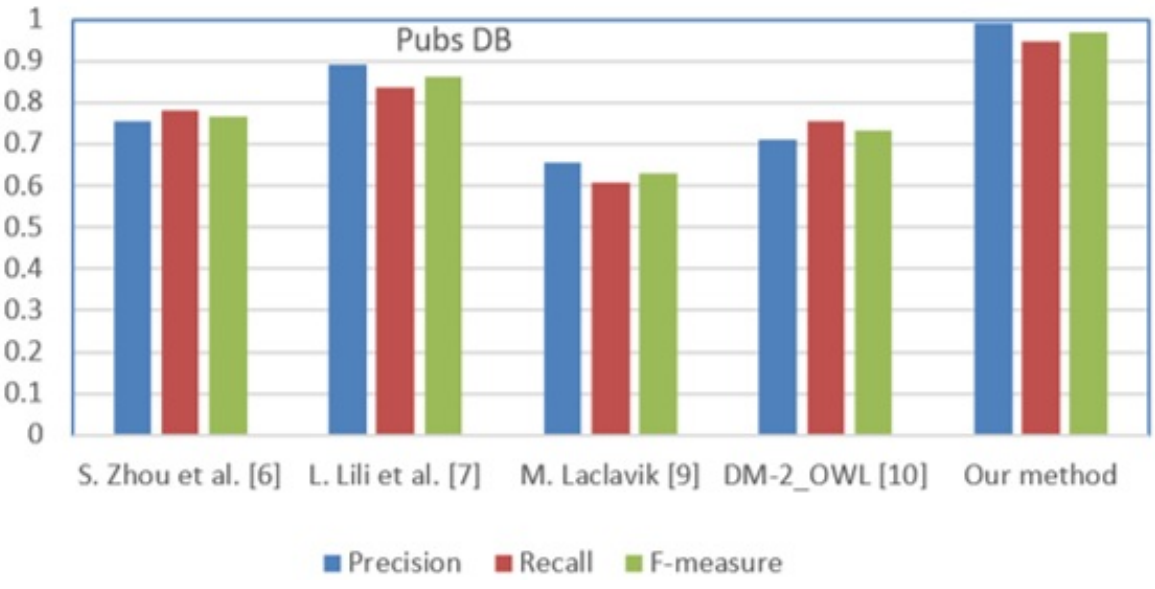

Figure 4. Matching comparison between our method and related work on Pubs database

keys and primary keys. In this experiment, the total number of the relationships in the Northwind database is higher than that of Pubs database. Therefore, for those methods which do not maintain the foreign key and primary key relationship, their matching results in the Northwind database are lower than those in the Pubs database. For instance, the F-measure score of the M. Laclavik [9] in the Fig. 3 is only 55% compared with 62% in the Fig. 4.

5. DISCUSSION AND CONCLUSION

Transformation from relational database into OWL ontology plays a critical role in realizing the Semantic Web as well as in many data sharing problems. There are many approaches mentioning this transformation. Moreover, most of those approaches do not discover the similarity of duplicate columns between tables. They simply provide each column an *rdf:ID*, this solution may lead to the data redundancy when those columns represent the same information.

Our study overcomes this problem by providing the similarity measure for duplicate column before transforming them into OWL ontology. Moreover, several approaches directly transform relational tuples into OWL triples without keeping the foreign key and primary key relationships. Other methods transform some relational tuples into the OWL individuals and do not consider the OWLS semantic constraints and relational datas structure. Our proposed RDB2OWL method can transform all data from the relations or can extract any required information while keeping the relationship between primary keys and foreign keys and improve the relational data semantics by using OWL vocabularies. The experimental results show that our proposed method outperforms other related work due to these reasons.

Finally, all the steps in our proposed method can be executed automatically without any human intervention. This algorithm can be also implemented as an intermediate module between any relational database and Semantic Web page. The extracted information can be selected by the users. Our future direction is to measure the similarity between relational database and the OWL ontology to find the appropriate matches between them.

REFERENCES

[1] B. He, M. Patel, Z. Zhang, K.C.-C. Chang, "Accessing the deep web", *Communications of the ACM*, vol.50, pp. 94–101, 2007.

[2] James Hendler, Tim Berners-Lee, Eric Miller, "Integrating applications on the Semantic Web", *Journal of the Institute of Electrical Engineers of Japan*, vol 122, no. 10, pp.676–680, 2002.

[3] Deborah L. McGuinness, Frank van Harmelen, "OWL Web ontology language overview", February 2004, http://www.w3.org/TR/owl-features/.

[4] Kamran Munir, M. Sheraz Anjum, "The use of ontologies for effective knowledge modelling and information retrieval", *Applied Computing and Informatic,* https://doi.org/10.1016/j.aci.2017.07.003

[5] Leila Zemmouchi-Ghomari, "Cohabitation of relational databases and domain ontologies in the Semantic Web context". *Journal of Systems Integration,* vol. 9, no. 1, pp. 42–57, 2018.

[6] Shufeng Zhou, Haiyun Ling, Mei Han, Huaiwei Zhang, "Ontology generator from relational database based on Jena", *Computer and Information Science*, vol. 3, no. 2, pp. 263–267, 2010.

[7] Lin Lili, Xu Zhuoming, Ying Ding, "OWL ontology extractions from relational databases via database reverse engineering", *Journal of Software*, vol. 8, no. 11, pp. 2749–2760, 2013.

[8] El Idrissi B., Baina S., Baina K, "Ontology learning from relational database: How to label the relationships between concepts", *Proceedings 11*th *International Conference, BDAS 2015*, Ustro, Poland, May 26-29, 2015,

[9] Michal Laclavik, "RDB2Onto: Relational database data to ontology individuals mapping", *Proceedings of Tools for Acquisition, Organisation and Presenting of Information and Knowledge*, Slovakia, 2006, pp. 86–89.

[10] K. M. Albarrak and E. H. Sibley, "Translating relational & object-relational database models into OWL models", *Proceedings of the 2009 IEEE International Conference on Information Reuse & Integration (IRI 2009)*, Las Vegas, NV, USA, Aug., 10-12, 2009 (pp. 336-341).

[11] Lei Zhang, Jing Li, "Automatic generation of ontology based on database, *Journal of Computational Information Systems*, vol. 7, no. 4, pp. 1148–1154, 2011.

[12] Juan Sequeda, Marcelo Arenas, Daniel P. Miranker, "On directly mapping relational databases to RDF and OWL", *Proceeding WWW '12 Proceedings of the 21st international conference on World Wide Web*, Lyon, France, April, 16-20, 2012 (pp.649–658).

[13] Guohua Shen, Zhiqiu Huang, Xiaodong Zhu, Xiaofei Zhao, "Research on the rules of mapping from relational model to OWL", *Proc. of OWLED06 Workshop on OWL: Experiences and direction*, 2006.

[14] I. Astrova, "Rules for mapping SQL relational databases to OWL ontologies", *Metadata and Semantics*, Springer, Boston, MA, 2009 (pp. 415–424).

[15] A. Buccella, M. R. Penabad, F. R. Rodriguez, A. Farina and A. Cechich, "From relational databases to OWL ontologies," *Proceedings of 6*th *Russian Conference on Digital Libraries (RCDL 2004)*, 2004.

[16] B. Motik, I. Horrocks and U. Sattler, "Bridging the gap between OWL and relational databases", *Journal of Web Semantics*, vol. 7, no. 2, pp. 74–89, April 2009.

[17] Z. Xu, S. Zhang and Y. Dong, "Mapping between relational database schema and OWL ontology for deep annotation", *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, IEEE Computer Society Washington, DC, USA, December, 18–22, 2006 (pp. 548-552).

[18] George A. Miller, "WordNet a lexical database for English", *Magazine Communications of the ACM CACM Homepage Archive,* vol. 38, no. 11, pp. 39–41, Nov. 1995.

[19] R. Rada, H. Mili, E. Bicknell, and M. Blettner, "Development and application of a metric on semantic nets", *IEEE Transactions on Systems, Man and Cybernetics*, vol. 19, no.1, pp.17–30, 1989.

[20] Ta Duy Cong Chien and Phan Thi Tuoi, "Building ontology based-on heterogeneous data", *Journal of Computer Science and Cybernetics*, vol.31, no.2, pp.149–158, 2015.

[21] Vo Hoang Lien Minh and Quang Hoang, "Transforming extended entity-relationship model into OWL ontology in temporal databases", *Journal of Science and Cybernetics*, vol.34, no.1, pp.77–96, 2018.

Received on September 16, 2018

Revised on Octorber 23, 2018