IJICTR International Journal of Information & Communication Technology Research

Volume 9 - Number 1- Winter 2017(17-23)

Inconsistency Repair to Improve the Alignment Results of Ontology Matchers

Bahareh Behkamal
Department of Computer Engineering
Ferdowsi University of Mashhad
Mashhad, Iran
b.behkamal@mail.um.ac.ir

Mahmoud Naghibzadeh
Department of Computer Engineering
Ferdowsi University of Mashhad,
Mashhad, Iran
naghibzadeh@um.ac.ir

Received: October 22, 2016- Accepted: March 14, 2017

***Abstract*— Ontology inconsistency is one of the most important topics in the field of ontology matching. Until now many matchers are introduced but most of them suffer from inconsistencies. Many of the ontology matching tools have severe problems with respect to the quality of matching results and therefore the results of matching process is not adequate. In this paper, we focus on this topic and present a new method to produce better results from the matching process. The major novelty of this paper is in detecting the inconsistencies in ontologies before starting the matching process. In this phase, many problems caused by ontology diversity are resolved. Besides, some new patterns and inconsistencies in ontologies are detected and then refactoring operations are applied on them. At the end, one of the well-known matchers in OAEI is selected to evaluate our work. Experimental results show that the transformed ontologies are more efficient than original unrepaired ones with respect to the standard evaluation measures.**

Keywords- Ontology matching; Alignment; Inconsistency; Refactoring; Pattern detection

I. INTRODUCTION

The vast progress of data and communication on the web has caused a huge amount of diversity in information. The problem of managing heterogeneity in various information resources is increasing. Until now many solutions have been proposed to facilitate this problem, and specifically, to automate integration of distributed data resources. Among them, semantic technologies have attracted particular attention. One of the best semantic technologies in this field is ontology matching. Ontology matching is a technique that takes the ontologies as an input and extracts the alignments as an output. The alignment is a set of correspondences between entities of ontologies that are semantically related. These correspondences can be used for various tasks, such as ontology integration [1], ontology evolution [2], data integration [3], and data warehouses [4]. Until now, many different tools for matching process are developed. However, most of them suffer from many problems with respect to the quality of matching results. Thus, in this paper we proposed an approach to get a better results from matching processes. The solution is adding a preprocessing phase to matchers. In the preprocessing phase, many input ontologies are analyzed in order to detect inconsistencies and inappropriate patterns modeled by various developers. For detecting these inconsistencies, ontology preprocessing language (OPPL) is used. Then, the refactoring rules are applied on detected patterns to repair the inconsistencies in input ontologies. At the end, assimilated ontologies delivered to matchers for matching process. In this paper for evaluating this work one of the best matcher namely, ASMOV from OAEI is selected. The Ontology Alignment Evaluation Initiative (OAEI) is a coordinated international initiative, which evaluate all matchers every year. As you can see in Figure 1, ASMOV has a good rank in comparison to other matchers with respect to the standard evaluation measurements such as precision, recall, and F-measure [5]. The precision, recall, and F-measure are explained in Section V. The second reason for selecting the ASMOV to evaluate our work is that it can do *n:m* alignment in contrast to other matchers like, SAMBO, Falcon, DSsim, RiMOM, Anchor-Flood, and AgreementMaker which can do only 1:1 alignment.

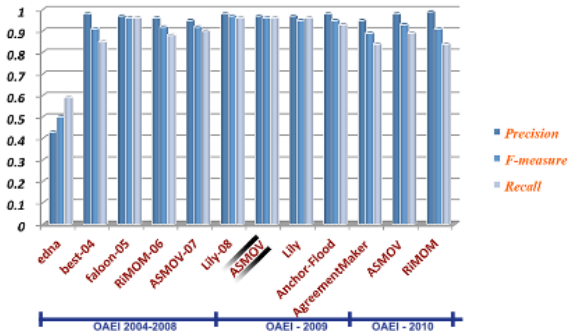

Fig. 1: Comparison of matching quality results for the duration of 2004- 2010 [5]

The rest of the paper is organized as follows: Section 2 introduces some general terminologies relevant to this research. Section 3 gives a theoretical background of related work and Section 4 elaborates on the methodology. In Section 5, the quality of the matching results is evaluated. Finally, Section 7 provides the conclusion and future work.

II. GENERAL TERMINOLOGIES

In the following some preliminary definitions and terms that are relevant to this literature and are used throughout this paper are described.

A. Ontology

Ontology *O* contains a set of entities related to a number of relations. Entities of an ontology can be divided into components as follows [6]:

- Classes (*C*): Classes define the concepts within the ontology
- Individuals (*I*): Individuals denote the object instances of classes
- Literals (*L*): Literals represent concrete data values
- Data types (*T*): Data types determine the possible types of those values
- Object properties (*OP*): Include the definitions of possible associations between two individuals
- Data type properties (*DP*): Include the definitions of possible associations between one individual and a literal.

There are four specific relations in ontology matching: equivalence, subsumption, disjointness, and membership.

B. Matching process

Matching is the process of finding the relations and correspondences between entities of different ontologies. The matching operation determines the alignment *A* for a pair of ontologies. Generally, matching algorithms can be classified based on the input of the algorithms, the characteristics of the matching process, and the output of the algorithms. The input dimension focuses on the input type on which algorithms operate. Algorithms can be classified depending on the data/conceptual models in which ontologies or schemas are described. The matching process can be based on its general properties. In particular, this depends on the approximate or exact nature of its computation. The output of a matching algorithm is related to the form of the alignment. For example, the correspondence between ontology entities is either one-to-one or not. Another dimension concerns is the kind of relations between entities that a system can provide. Most of the matching softwares focus on equivalence (=) relation, while a few others are able to provide more expressive results (e.g. subsumption and incompatibility) [7, 8].

C. Alignment

The alignment of ontologies o and o' is a set of correspondences between two or more (in the case of multiple matching) ontologies. The alignment is the output of the matching process between the entities of o and o'. The alignment can be achieved in various cardinalities: *1:1* (one-to-one), *1:m* (one-to-many), *n:1* (many-to-one) or *n:m* (many-to-many).

D. Refactoring

Refactoring is recognized as changes that are made to the internal structure of the software in order to make it easier to understand and to modify without changing its observable behavior.

III. THEORETICAL BACKGROUND

To establish suitable semantic correspondences between entities of different ontologies, the integration of the input ontologies is needed. Unfortunately, many ontology matching systems ignore the semantics of the input ontologies in the matching process. Therefore, the matching result is not satisfiable. In this paper, we attempt to combine four apparently distant areas to handle this problem. These areas are: ontology matching, ontology patterns, ontology refactoring, and inconsistency repair. Accordingly, in this section, some research conducted in each of these areas are described.

Research in ontology matching has been burgeoning since the early 2000's. So far, most articles on the ontology matching field have focused on the method of matching processes and have introduced some matchers with diverse approaches. In this section, some matchers which have high ranks in ontology alignment evaluation initiative (OAEI) are introduced. ASMOV (Automated Semantic Matching of Ontologies with Verification) [6] have been applied to the lexical and structural characteristics of two ontologies to calculate the similarity measures. Then the alignment have been verified to ensure that it does not contain semantic inconsistencies. RiMOM [9] is a dynamic multi-strategy ontology alignment framework that combines multiple strategies to improve matching efficiency. The key intuition in this framework is that similarity characteristics between ontologies may vary widely. This approach has considered both the textual and structural characteristics of ontologies. RiMOM is a framework based on risk minimization of the Bayesian

decision systems. It employs multiple ontology alignment strategies and sets a combination weight. Another system is Falcon-AO [10], a practical ontology matching system with good performance that acts based on a number of remarkable features. It is an automatic ontology matching system that uses multiple elementary matchers (V-Doc, GMO and PBM), coordination rules, and the similarity combination strategy. PROMPT [11] algorithm consists of an interactive ontology merging tool and a graph-based mapping called Anchor-PROMPT. Anchor-PROMPT [12] uses linguistic "anchors" as a starting point and analyses these anchors in terms of the structure of the ontologies. GLUE [13] discovers mappings through multiple learners that analyze the taxonomy and the information within concept instances of ontologies. S-Match [14] is a deductive technique for semantic ontology matching which employs a number of elemental level matchers to express ontologies as logical formulas and then use a propositional satisfiability (SAT) solver to check the validity of these formulas.

Generally, all of the above matching algorithms are classified into two categories: elemental and structural. Elemental level matching techniques compute matching elements by analyzing entities in isolation and ignoring their relations with other entities. Structural level techniques compute matching elements by analyzing how entities appear together in a structure and considering the relation of concepts in taxonomy tree [15].

In recent years, some works on ontology patterns is done [16-19]. Ontology patterns have been used in many fields, but they have rarely been applied in the field of ontology matching. Ontology patterns are mainly inspired by software engineering and knowledge engineering [20]. In the following, some previous works in the field of ontology matching by considering the ontology patterns is described. The paper in [21] involves testing the impact of ontology refactoring on the results of three matcher, namely HMatch, Falcon-AO, and ASMOV. In this paper, some modeling errors via name structure analysis were found and three refactoring operations were applied. By considering semantic structures, authors in [22] analyzed collections of OWL ontologies in order to determine the number of occurrences of several combined name and graph patterns. These structures ranged from simple subsumption to more complex constructions. The goal of this paper is to facilitate automatic alignment among different models by finding such patterns in the given ontologies. In [23], the authors concentrate on detection and mutual matching of semantic structures in ontologies. The authors use the equivalence relation, as well as analyzing homogeneous correspondence. Research in [24] presents a simple method of tracking name patterns over OWL ontology taxonomies. This method helps to detect several probable taxonomic errors and modeling inconsistencies with respect to their set-theoretic interpretations. In [25] authors applied weights to the edges of WordNet hierarchy to improve the semantic word similarity. Furthermore the distance of two words and depth of words in semantic similarity assessment are utilized. This approach can be applied for inconsistencies detection phase of matching process.

Until now ontology refactoring is employed in many different areas [26-29], but the impact of ontology refactoring on the ontology matching field is rarely discussed [21, 23, 30]. In this paper, we focus on this matter. In [26], the authors focus on the detection of anomalies as an important criterion for verification. In this paper, some approaches for the syntactic verification of ontologies are explained and definitions are extended with respect to the existence of rules. Furthermore, novel measures are introduced for detecting the parts of the ontology that may create problems for maintainability. This paper [27] proposed an approach for refactoring multimodal knowledge on the basis of a generic data structure in order to support the representation of multimodal knowledge. Moreover, how this data structure was created from given documents (i.e. the most general mode of knowledge) was explained, along with how different refactoring could be performed by considering various levels of formality. In [29], the authors present the semantic knowledge wiki, Know WE, used to capture and share ontological knowledge for the effective elicitation of problem solving knowledge. Also, a distributed knowledge acquisition process and refactoring phase are shown. In [30], a semi-automatic process for lifting meta-models into ontologies is proposed that allows creating the semantic integration of modeling languages. In so doing, implicit concepts in the meta-model are changed to explicit concepts in the ontology. The application of refactoring patterns on the resulting ontologies could improve automation support for semantic integration tasks. The paper [28] presents a method to develop conceptual schemas as refinements of more general ontologies. For obtaining final conceptual schemas, three activities are performed: refinement, pruning, and refactoring. The refinement phase is done to execute a set of additive operations to the ontology to create necessary elements. Afterwards, in the pruning phase, some unnecessary elements are deleted. Then, a pruned ontology is obtained. At the end, the pruned ontology can be improved by using refactoring operations to obtain the final conceptual schema.

IV. METHODOLOGY

In this part, an approach is proposed to improve the quality of the matching results. The aim of this approach is improving the alignment results by finding the inconsistencies before matching process. Our previous works [31], [32] focus on only lexical and structural patterns, but in this work we concentrate on some new inconsistency patterns. To accomplish this aim, a pre-processing phase is added to matchers. In the pre-processing phase, at first, a comprehensive survey to find the inconsistencies in input ontologies are performed. Then various lexical and structural patterns, which have been modeled by different developers, are detected. Afterward, some refactoring operations are applied on these patterns for repairing

the ontologies. Finally, these repaired ontologies are used as inputs of the matching process. This process is evaluated by ASMOV [33]. Experimental results indicate that better outcomes can be achieved by applying the pre-processing phase as opposed to original ones. In the following the details of work is elaborated.

A. *First step: Inconsistency Detection Phase*

In this step, some inconsistencies were detected based on our preliminary analysis of many ontologies. For detecting lexical inconsistencies, the name of entities, especially classes in OWL ontologies are analyzed. The lexical feature consists of all information readable by humans in the ontology. Various ontologies use different methods for defining the names of homogeneous concepts, especially for compound words. In OWL ontologies, different styles in concept naming lead to many obstacles for calculating lexical similarities in matchers.

In ASMOV, three lexical concepts in OWL ontologies are considered: id, label, and comment. ASMOV uses the Lin method [34] for calculating the lexical similarity. As an instance, in two ontologies of a conference track, namely *Conference* and *Ekaw* two different class naming for a similar concept is discovered, <Conference#conference-www> ~ <Ekaw#website> and also <Conference#rejected-contribution> ~ <Ekaw#rejected-paper >, both of them couldn't be found by lexical similarity phase of ASMOV. Therefore, to solve these kinds of problems, some lexical patterns and inconsistencies are detected based on naming ontology design patterns [35] for the purpose of unifying the naming for these different styles of naming. To accomplish this, we used one refactoring operation called renaming operation (RN), which is described in the next section. By doing this, calculating the lexical similarity in matchers, which is done by different methods, can do better than before. Thus, better results can be obtained from the matching process.

Structural patterns are based on the fact that the taxonomic structures of ontologies are often varied and confusing. One reason for this is that different developers have dissimilar viewpoints for developing ontologies. Therefore, they utilize different hierarchies and granularities for defining the entities of ontologies in the same domain. For example, in two ontologies of the conference track namely *Conference* and *Ekaw*, realize that there are two different granularities in concept naming for the similar concept "*author*". In *Conference*, three levels of granularity for "*author"*is found which include: contribution_regular-author, contribution_co-author, and Conference _1th-author. However, in *Ekaw*, there was only one level of granularity for author, namely Paper_author. Furthermore, many problems for calculating the relational similarity by some matchers have been recognized. The relational or hierarchical similarity phase in most matchers is computed by combining the similarities between the parents and children of entities that want to be compared. By considering the problems mentioned above and matcher's work, we realized that different taxonomic structures and different granularities in peer ontologies cause many problems in the matching process. For solving this problem, another refactoring operation, called restructuring operation (RS), is employed for assimilating the structural features of OWL ontologies. Our results show that, in most ontologies, there are significant number of occurrences of the aforementioned patterns.

B. *Second step: Refactoring phase*

In this phase, by refactoring operations some patterns and inconsistencies, which are detected in previous phase, are repaired. All cases of the modeling errors detected via some patterns mentioned earlier can be repaired by two refactoring operations. The detection of these patterns is the starting point for a refactoring. Generally, refactoring is a process for performing some changes in the internal structure of the software in order to make it easier to understand and to modify without changing its discernible behavior. In this literature, the refactoring process of an ontology matching field is applied. Thus, some changes are done in ontologies by a semi-automatic process. By doing this, new and more understandable versions of ontologies for users and matchers are produced. These versions of ontologies can be utilized more effectively by different ontology matching tools.

There are three general refactoring operations: adding operation (ADD), restructuring operation (RS), and renaming operation (RN). These operations consist of different steps depending on the detected situation [21]. In this paper, RN and RS are used for lexical patterns and structural patterns, respectively. More desirable results in lexical similarity of matchers can be obtained by applying the rename operations for the name of the classes. The rename operations are done by considering the name of the classes in the ontology that have the same taxonomic structures in the peer ontology. Furthermore, by considering the parent-child relations and various granularities used in peer ontologies, restructuring operations are applied for assimilating the structural features of the OWL ontologies. Experimental results show that, better results can be achieved from the structural similarity phase of matchers by transforming a part of ontology into another one. We carry out our experiments on seven pairs of ontologies from the conference track. The reason for choosing these seven pairs among other ontologies is described in the next section.

The number of RN and RS operations applied on these seven pairs of ontologies is explained in the following. In four pairs of ontologies, <*Cmt- ConfOf*>, <*Cmt-Ekaw*>, <*Conference-Ekaw*>,and <*Edas-Ekaw*>, RN operations are applied more than RS operations, because of the many different lexical patterns find in these pairs. Besides, in other ontology pairs, <Cmt-Sigkdd>, <Conference-ConfOf>, and <ConfOf-Sigkdd>, RS operations are utilized more than RN operations, because these pairs of ontologies have different hierarchical structures and RS operations is used for assimilating the taxonomies.

C. Data set

Some ontologies from OAEI is selected to evaluate this work. The OAEI offers several tracks and subtracks concentrated in different types of matching problems. Our approach was tested on the *Conference Track* [36]. They are described in OWL-DL and published in the RDF/XML format [37]. This data set is a well-known data set to the organizers and has been used in many ontology matching evaluations. The *Conference* dataset can be viewed as a much more challenging test cases in contrast to other ontologies of OAEI, such as the *Benchmark* dataset [38], [39]. Our experiment was carried out on six out of sixteen ontologies of the *Conference Track*. These ontologies are *cmt*, *confOf*, *ekaw*, *conference*, *edas,* and *sigkdd*. The reason for selecting these six ontologies among others is that reference mapping (also referred to as the gold standard) is available for all possible combinations of these selected ontologies. To evaluate the accuracy of the matching process, it is necessary to determine both the number of correctly found correspondences and the number of incorrectly found correspondences.

D. Implementation

Our implementation is based on the employment of Java language with Jena API in Net Beans IDE. Furthermore, protégé and the Ontology Pre-Processor Language (OPPL) were used for manipulating ontologies written in OWL. OPPL is a domain-specific language, based on the Manchester OWL Syntax. OPPL instructions can add or remove entities and add/remove axioms to entities in OWL ontology. The OPPL Instruction Manager is a Java library that processes OPPL instructions to make changes in OWL ontology. This language is also suitable for defining independent modeling macros that can be applied across ontologies [40].

E. Practical Example

Presented in this section is a practical example to clarify the proposed approach by testing the work with ASMOV matcher. Figure 2 illustrates different styles in class naming and various taxonomic structures for defining the same concepts in a part of two ontologies, namely ConfOf and Sigkdd.

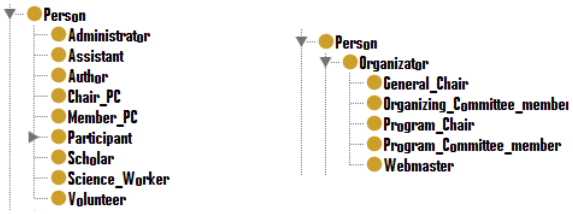

Fig. 2: Different class naming and taxonomic structures using Protégé software [41]

The inconsistencies are located in two peer ontologies are detected with OPPL. After that the ontologies are manipulated by applying the refactoring rules on each one. Then assimilated ontologies are delivered to ASMOV matcher as inputs. The alignment results are shown in figure 3.

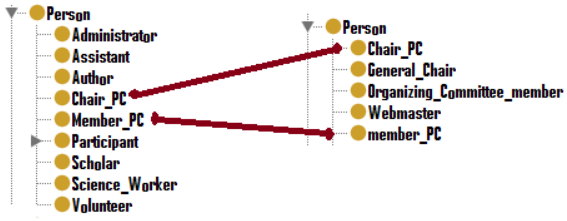

Fig. 3: Correspondences found by ASMOV after refactoring [6]

V. EVALUATION OF THE MATCHING RESULTS

For evaluating the matching results three standard measures, precision, recall, and F-measure is used. Precision is defined as the number of correctly found correspondences divided by the total number of found correspondences. Recall is considered as the number of correctly found correspondences divided by the number of reference alignment. A perfect precision score of 1.0 means that every correspondence computed by the algorithm was correct (correctness), whereas a perfect recall scores of 1.0 means that all correct correspondences were found (completeness).

Precision and recall are defined in (1), (2) [42].

$$\text{Precision} = \frac{\#\text{Correctly found matches}}{\#\text{Number of all found matches}} \quad (1)$$

$$\text{Recall} = \frac{\#\text{ Correctly found matches}}{\#\text{Number of reference alignment}} \quad (2)$$

F-measure represents a trade-off between precision and recall and it is calculated as (3).

$$\text{F} - \text{Measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (3)$$

Experiments were performed on seven pairs of ontologies from the conference track. The alignments generated automatically by ASMOV for these pairs of ontologies before and after of applying the proposed approach. The results were illustrated in figure 4, figure 5, and figure 6. The results of our experiments show that transformed ontologies improve the matching results with respect to the standard evaluation measures i.e. precision, recall, and F-measure.

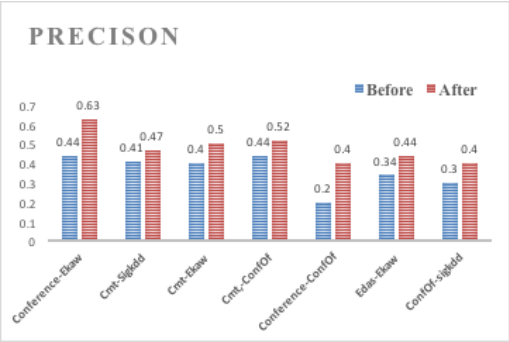

Fig. 4: The Effect of the method on the precision

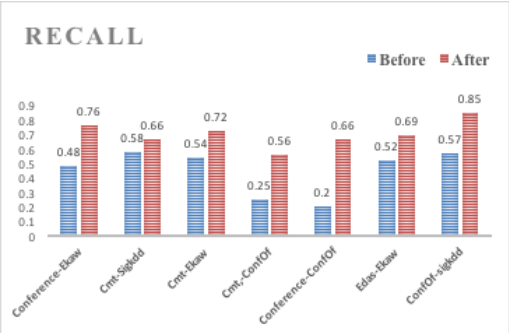

Fig. 5: The effect of the method on the Recall

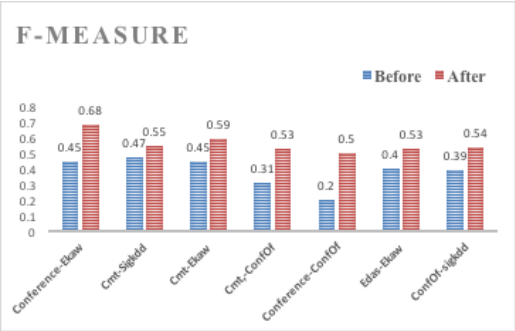

Fig. 6: The effect of the method on the F-Measure

VI. CONCLUSIONS AND FUTURE WORKS

In this paper, an approach has been presented for overcoming the uncertainty in the ontology matching results. This work is done by detecting the inconsistencies in ontologies before starting the matching process. In the first step, many problems caused by ontology diversity are resolved. So that, some inconsistencies and unexpected errors, which have been modeled in input ontologies, are detected. After that two refactoring operations, RN and RS, is applied to repair them. This work makes transformed ontologies easier to understand by both humans and matchers. Furthermore, some common mistakes in the alignment results are reduced. The transformed ontologies evaluated with one of the best-ranked matchers, ASMOV. Our experiments were carried out on ontologies of the *conference track*. Experimental results show that our approach improved the quality of the matching process with respect to standard evaluation measurements, i.e. precision, recall, and F-measure.

For future research, new solutions can be proposed for overcoming the uncertainty and other challenges in the field of ontology matching. Furthermore, our approach can be tested on other matching tools, especially those participating in the OAEI contest. Moreover, some detectable patterns for discovering errors of ontologies and other refactoring operations for repairing them can be extended.

ACKNOWLEDGEMENT

This research was partially supported by Ferdowsi University of Mashhad under the grant number 2/39961 to which the authors likes to extend his thanks to.

REFERENCES

[1] A. Isaac, S. Wang, C. Zinn, H. Matthezing, L. van der Meij, and S. Schlobach, "Evaluating Thesaurus Alignments for Semantic Interoperability in the Library Domain," IEEE Intelligent Systems, vol. 24, no. 2, pp. 76-86, Mar./Apr. 2009.

[2] N.F. Noy, A. Chugh, W. Liu, and M.A. Musen, "A Framework for Ontology Evolution in Collaborative Environments," Proc. Fifth Int'l Semantic Web Conf. (ISWC), pp. 544-558, 2006.

[3] P.P. Talukdar, Z.G. Ives, and F. Pereira, "Automatically Incorpor- ating New Sources in Keyword Search-Based Data Integration," Proc. 29th ACM SIGMOD Int'l Conf. Management of Data (SIGMOD), pp. 387-398, 2010.

[4] S. Dessloch, M.A. Hernández, R. Wisnesky, A. Radwan, and J. Zhou, "Orchid: Integrating Schema Mapping and ETL," Proc. 24th Int'l Conf. Data Eng. (ICDE), pp.1307-1316, 2008.

[5] Shvaiko, Pavel, and Jérôme Euzenat. "Ontology matching: state of the art and future challenges", IEEE Transactions on Knowledge and Data Engineering, pp. 158-176, 2013.

[6] Jean-Marya, Y.R., Shironoshitaa, E.P. & Kabuka, M. R. "Ontology matching with semantic verification", *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 7, pp. 235-251, 2009.

[7] Euzenat, J. & Shvaiko, P, "*Ontology matching*". Berlin: Springer, pp. 51-54, 2007.

[8] Shvaiko, P. & Euzenat, J."A survey of schema-based matching approaches", *Journal on Data Semantics IV*. vol. 6, pp. 146-171, 2005.

[9] Li, J., Tang, J., Li, Y. & Luo, Q. "RiMOM: A dynamic multistrategy ontology alignment framework". IEEE Transactions on Knowledge and Data Engineering, vol. 21, pp. 1218-1232, 2009.

[10] Hu, W. & Qu, Y. "Falcon-AO: A practical ontology matching system", Journal of Web Semantics: Science, Services and Agents on the World Wide Web, vol. 6, pp. 237-239, 2008.

[11] Noy, N. F. & Musen, M. A. "PROMPT: algorithm and tool for automated ontology merging and alignment", Proceedings of the 17th International Conference on Artificial Intelligence (AAAI), pp. 450-455, 2000.

[12] Noy, N. F. & Musen, M. A. "Anchor-PROMPT: using non-local context for semantic matching", In Workshop on Ontologies and Information Sharing at the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI), pp. 63-70, 2001.

[13] Doan, A., Madhavan, J., Domingos, P. & Halevy, A. "Ontology matching: a machine learning approach", Handbook on Ontologies, Berlin, Springer-Verlag, pp. 385-404, 2004.

[14] Giunchiglia, F., Shvaiko, P. & Yatskevich, M, "S-Match: an algorithm and implementation of semantic matching"

Proceedings of the first European Semantic Web Symposium (ESWS), vol. 3053, pp. 61-75, 2004.

[15] Shvaiko, P. & Euzenat, J. "A survey of schema-based matching approaches". Journal on Data Semantics *IV*. vol. 6, pp. 146-171, 2005.

[16] Šváb-Zamazal, Ondřej, et al. "Detection and Transformation of Ontology Patterns", Knowledge Discovery, Knowlege Engineering and Knowledge Management. Springer Berlin Heidelberg, pp. 210-223, 2011.

[17] Šváb-Zamazal, Ondřej, Vojtěch Svátek, and Luigi Iannone. "Pattern-based ontology transformation service exploiting OPPL and OWL-API", *Knowledge Engineering and Management by the Masses*. Springer Berlin Heidelberg, pp. 105-119, 2010.

[18] Scharffe, François, Ondřej Zamazal, and Dieter Fensel. "Ontology alignment design patterns", Knowledge and information systems , vol. 40, no.1, pp. 1-28, 2014.

[19] Šváb-Zamazal, Ondrej, et al. "Tools for Pattern-Based Transformation of OWL Ontologies", *Presented as Demo at ISWC* (2011).

[20] Clark, P., Thompson, J. & Porter, B. W. "Knowledge patterns", Proceedings of the international Conference on Principles of Knowledge Representation and Reasoning, pp. 591-600, 2000.

[21] Sváb-Zamazal, O., Svátek, V., Meilicke, C. & Stuckenschmidt, H. "Testing the impact of pattern-based ontology refactoring on ontology matching results", Proceedings of the ISWC 2008 Workshop on Ontology Matching, pp. 229-233, 2008.

[22] Svab-Zamazal, O. & Svatek, V. "Analysing ontological structures through name pattern tracking", Proceedings of the 16th International Conference on Knowledge Engineering and Knowledge Management, Springer LNCS,pp. 213-228, 2008.

[23] Svab-Zamazal, O. & Svatek, V. "Towards ontology matching via pattern-based detection of semantic structures in OWL ontologies", Proceedings of the Znalosti Czecho-Slovak Knowledge Technology conference, 2009.

[24] Svab-Zamazal, O. & Svatek, V. "Tracking name patterns in OWL ontologies", Proceedings of the 5th International EON Workshop Located at the 6th International Semantic Web Conference (ISWC), pp. 61-70, 2007.

[25] Ahsaee, M.G., Naghibzadeh, M. and Naeini, S.E.Y., 2014. Semantic similarity assessment of words using weighted WordNet. International Journal of Machine Learning and Cybernetics, vol.5, no.3, pp.479-490.

[26] Baumeister, J. & Seipel, D. "Verification and refactoring of ontologies with rules", Proceedings of the 15th International Conference on Knowledge Engineering and Knowledge Management (EKAW), pp. 82-95, 2006.

[27] Reutelshoefer, J., Baumeister, J. & Puppe, F. "A data structure for the refactoring of multimodal knowledge", Proceedings of the 5th Workshop on Knowledge Engineering and Software Engineering, pp. 33-45, 2009.

[28] Caralt, J. C. "Ontology-driven information systems: pruning and refactoring of ontologies", Proceedings of the Doctoral Syposium of 7 th International Conference on the Unified Modeling Language, 2004.

[29] Baumeister, J., Reutelshoefer, J., Haupt, F. & Nadrowski, K. "Capture and refactoring in knowledge wikis, Proceedings of the 2nd Workshop on Scientific Communities of Practice, 2008.

[30] Kapsammer, E., Kargl, H., Kramler, G., Reiter, T., Retschitzegger, W. & Wimmer, M. "Lifting metamodels to ontologies - a step to the semantic integration of modeling languages", Proceedings of the ACM/IEEE 9th International Conference on Model Driven Engineering Languages and Systems, pp. 528-542, 2006.

[31] Bahareh Behkamal, Mahmoud Naghibzadeh, and Reza Askari Moghadam."Using pattern detection techniques and refactoring to improve the performance of ASMOV", 5th International Symposium on Telecommunications (IST 2010), pp. 979-984, 2010.

[32] Bahareh Behkamal, Mahmoud Naghibzadeh, and Reza Askari Moghadam."Adding a Pre processing Phase to ASMOV for Improving the Alignment Result", International Journal of Advanced Science and Technology, Vol. 49, pp. 25-36, 2012.

[33] Kabuka, Mansur. "Ontology alignment with semantic validation", U.S. Patent No. 8,738,636. 27 May 2014.

[34] D. Lin, "An information-theoretic definition of similarity", in presented at Proceedings of the 15th International Conference on Machine Learning (ICML), Morgan Kaufmann, San Francisco, Vol.98, pp.296-304, 1998.

[35] N. Noy and D. L. McGuinness, "Ontology development 101: A guide to creating your first ontology", in Technical report, Stanford Knowledge Systems Laboratory, (2001).

[36] OAEI (2015). Ontology Alignment Evaluation Initiative. URL:<http://oaei.ontologymatching.org/2015/conference/index.html>

[37] O. Svab, V. Svatek and H. Stuckenschmidt, "A study in empirical and `casuistic' analysis of ontology mapping results", in In Proceedings of the 4th European conference on The Semantic Web (ESWC-07), Berlin, Heidelberg: Springer-Verlag, 2007.

[38] Alessandro Solimando, Ernesto Jiménez-Ruiz, Giovanna Guerrini: Detecting and Correcting Conservativity Principle Violations in Ontology-to-Ontology Mappings. International Semantic Web Conference (2), PP. 1-16, 2014.

[39] Alessandro Solimando, Ernesto Jiménez-Ruiz, Giovanna Guerrini: A Multi-strategy Approach for Detecting and Correcting Conservativity Principle Violations in Ontology Alignments. OWL: Experiences and Directions Workshop (OWLED 2014), PP. 13-24, 2014.

[40] Egaña, M., Stevens, R. & Antezana, E. "Transforming the axiomisation of ontologies: the ontology pre-processor language", Proceedings of the 4th International Workshop, OWL: Experiences and Directions (OWLED), 2008.

[41] Musen, M.A. The Protégé project: A look back and a look forward. AI Matters. Association of Computing Machinery Specific Interest Group in Artificial Intelligence, 1(4), June 2015. DOI: 10.1145/2557001.25757003

[42] Houshmand, M., Naghibzadeh, M. & Araban, S. Reliability-based similarity aggregation in ontology matching. Proceedings of the IEEE International Conference on Intelligent Computing and Intelligent Systems (ICIS 2010), pp. 744-749.

Bahareh Behkamal has received her B.S degree in Computer Engineering from Azad University of Mashhad, Iran in 2005 and her M.S degree in Computer Engineering from Pnu University of Tehran, Iran in 2011. She is currently Ph. D candidate in Computer Engineering at Ferdowsi University of Mashhad, Iran. Her past research interests include Semantic Web and Ontology Matching. She has published some papers in international journals and conferences. She is currently researching in the area on Structural Bioinformatics.

Mahmoud Naghibzadeh has received his M.Sc. and Ph.D. degrees in Computer Science and Computer Engineering, respectively, from University of Southern California (USC), USA. He is now a full professor at the Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran. He is the director of Knowledge Engineering Research Group (KERG) laboratory. His research interests include the scheduling aspects of real-time systems, Grid, Cloud, Multiprocessors, Multicores, and GPGPUs. He is also interested in Bioinformatics computer algorithms, especially protein tertiary structures and protein-protein interactions.

This Page intentionally left blank.