Falcon-AO++: An Improved Ontology Alignment System

Fatsuma Jauro
Department of Mathematics
Ahmadu Bello University, Zaria.

S.B. Junaidu
Iya Abubakar Computer Center,
Ahmadu Bello University, Zaria

S.E. Abdullahi
Department of Mathematics
Ahmadu Bello University, Zaria.

ABSTRACT

With the semantic web, data becomes machine-readable and ontologies define the data. Ontologies in any domain are heterogeneous due to rapid increase in ontology development and differences in views of developers. Agents can fully understand the data only if the correspondences between ontologies are known. Various ontology alignment systems have been developed to automatically discover such correspondences. However, human involvement is still indispensible because the results provided by fully automatic systems are not always complete or precise. This paper introduces Falcon-AO++, an extension of the Falcon-AO alignment system that supports the interactive contribution of a domain expert in the matching process. The evaluation results have shown that contribution of an expert and matching ability of matchers can improve alignment results.

General Terms

Ontology alignment

Keywords

Ontology matching, user input

1. INTRODUCTION

In the semantic web, ontologies describe domain by defining terms and relations enabling machines to understand the meaning and reason about data. Numerous ontologies have been developed by different designers to describe the same domain. These ontologies are heterogeneous due to variation in views of their developers. Ontology heterogeneity could be due to variation in used terms, depth or area of coverage etc. Ontology heterogeneity causes the problem of variation in meaning or ambiguity in terms interpretation. This is known as *semantic heterogeneity.*

Overcoming semantic heterogeneity is typically achieved in two steps [1]: (i) matching entities to determine an alignment and (ii) interpreting an alignment according to application needs, such as data translation or query answering. This paper focuses on ontology matching.
Ontology alignment or matching tries to solve semantic heterogeneity problems by means of discovering a map between similar terms (entities) of two different ontologies of the same domain. This enables applications using the ontologies to understand information and interoperate.

Different systems have been developed to automatically handle alignment process such as AOAS [2], OMReasoner [3], RiMOM [4], CIDER [5] and Falcon-AO [6]. However, the challenges faced by fully automatic methods are manifold, including vocabulary differences (e.g., due to synonymy and homonymy), modeling differences (e.g., due to different model granularity or different attribute formats) and different points of view on the modeled reality [7]. For many real-world datasets, fully automatic state-of-the-art tools still yield results at a quality level that is unsatisfying for many use cases [8]. Different researchers pointed out the need to involve humans in alignment process. According to Sarasua *et al* [9], Ontology alignment is still one of those problems that we cannot automate completely, and having a human in the loop might increase the quality of the results of machine-driven approaches.

Again Paulheim *et al.*, [8] stated that there is an upper bound to the quality of the alignment which is hard to exceed by fully automatic ontology matching tools. Furthermore, ontology matching is "a very challenging problem for both man and machine" which calls for semi-automatic approaches combining the strengths of automatic matching algorithms and the expertise of domain experts in the matching process.

In this paper, Falcon-AO++, a semi-automatic ontology alignment system that combines the knowledge of domain experts and the matching ability of matchers to align ontologies is introduced. The system is an extension of the Falcon-AO automatic ontology alignment system [6].

The paper is organised as follows: review of some ontology alignment systems is given in Section 2, architecture of Falcon-AO++ in Section 3, implemented processes in Section 4, evaluation and results in Section 5, and directions for future improvement in Section 6.

2. RELATED WORK

Different tools have been developed to align ontologies. Most of the systems apply different matching techniques. **AOAS** [2],[10] is a domain-specific ontology matching system developed specifically to align anatomical ontologies. It uses different techniques and an external resource; Unified Medical Language System (UMLS). The anatomy track of OAEI (Ontology Alignment Evaluation Initiative) proved that domain specific alignment systems perform better than domain independent systems [11]. However such systems fail in other domains. Falcon-AO++ is a domain independent system. **CHIMAERA** [12] semi-automatically aligns and merges ontologies. The system requires user intervention at almost every stage which adds more stress to the user. Falcon-AO++ only requires users to provide input information. **OMReasoner** [3] automatically performs ontology alignment using string technique and WordNet. It also performs reasoning about ontologies but does not consider structural information which is very important especially when concept names are meaningless. Falcon-AO++ uses structural information to match entities. **RiMOM** [4] is an alignment tool that employs different techniques to match ontologies at about six different stages. The system can handle large ontologies but it takes large amount of memory and very long time. Falcon-AO++ uses portioning technique for large ontologies which minimizes the amount of memory consumed and execution time. **ONION** [13] resolves terminological heterogeneity in ontologies and produces articulation rules for mappings. Similar to Falcon-AO++, it has a GUI and a human expert is involved. In ONION, the expert chooses, deletes, or modifies suggested matches using the GUI tool while in

Falcon-AO++ a user only gives input information. **Falcon-AO** [6] is a fully automatic and domain independent ontology alignment system. It uses string and structural techniques to align ontologies. It also uses partitioning technique for large ontologies and employs the idea of virtual documents. It considers both comment and label information attached to entities. The major drawbacks of this system are: 1. When the structures of given ontologies are dissimilar some alignments are lost. 2. The system accepts matches with high similarity from a string matcher, this leads to error alignment when equal terms are used to mean different things.

Falcon-AO was extended with user-input strategy so that domain experts can assist the system by providing input information. The aim is to improve results produced by Falcon-AO. An evaluation of the two systems; Falcon-AO and Falcon-AO++ was carried out using standard benchmarks to examine the impact of user interaction on alignment results.

3. ARCHITECTURE OF FALCON-AO++

The semi-automatic approach implements the idea that a domain expert can provide input information to an alignment system. Figure 1 presents the architecture of Falcon-AO++ which is an extension of the architecture of Falcon-AO [14]. In the system architecture, all the components apart from the integrated *User Restrictions* are components of Falcon-AO [14]. Brief description of the components is given below:

3.1 User Restriction

The system receives two ontologies as input. *User input* is generated by the *User Restrictions* component in the following stages:

3.1.1 Parsing and Entity Retrieval Component

The parsing and entity retrieval component parses input ontologies and retrieves all the entities in the ontologies to provide easy access.

3.1.2 Constraint Specification

This component allows a user to select pairs of entities and a constraint for each pair. Two constraints are supported in this paper;

3.1.2.1 Equivalent Constraint

This constraint allows two entities to be defined as equivalent. The constraint here prevents possible alignment lost.

3.1.2.2 Disjoint Constraint

This constraint allows two entities to be defined as entirely different even if their strings are exactly the same. The constraint also prevents possible error alignments.

3.1.3 Rule Implementation

The rule implementation component defines and implements a rule by *Jena* based on specified constraint.

3.2 Model Pool

In the model pool, the *ontology parser* checks the validity (parses) of given ontologies and creates a *model* (an internal representation) of the ontologies. The *model coordinator* adjusts the models using some coordination rules.

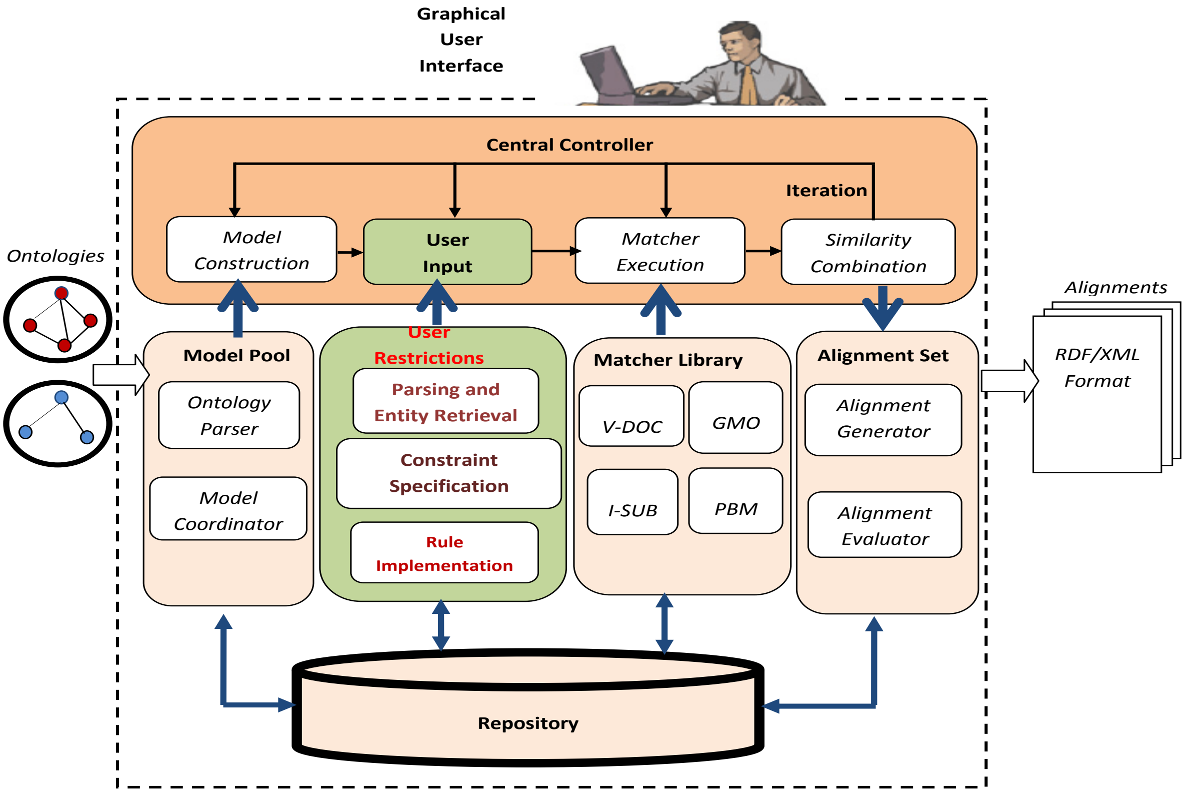


Figure 1: Architecture of Falcon-AO++

3.3 Matcher Library

The Matcher library Controls four different matchers used by the system. The matchers are described below.

3.3.1 PBM (Partition Block Matcher)

PBM [15] uses the *Divide and Conquer* approach to match ontologies. It is employed when the ontologies to be matched are large. The matcher first partitions entities of each input ontology into clusters based on their proximities in the graph, generates blocks by reassigning the link between entities, matches similar blocks from the two ontologies by distributing *anchors*, and finally matched blocks are passed to V-DOC and GMO for final alignment and output.

3.3.2 I-SUB

The ***I-SUB*** [14] matcher takes names of entities as string of characters and computes the similarity between the strings. The matcher has the capability of comparing not only the commonalities between strings but also their differences.

3.3.3 V-DOC (Virtual Document)

V-DOC [16] matcher is based on the idea that the meaning of an entity is encoded in its documentation. It therefore creates virtual documents for all the entities. The virtual document of an entity contains the entity's name and any description attached to it and also the names of its neighbours. The similarity between two entities comes from the existence of shared word(s) in their virtual documents.

3.3.4 GMO (Graph Matching for Ontologies)

GMO [17] uses the structural matching approach. It uses RDF bipartite graph to represent ontologies and computes structural similarities between domain entities and between statements (triples) in ontologies by recursively generating similarities in the bipartite graphs. It receives alignments found by V-DOC and I-SUB as external input and outputs additional alignments.

3.4 Central Controller

The Central Controller Controls the execution of matchers and other processes.

3.5 Alignment Set

In the Alignment Set, the *alignment generator* generates alignments, and the *alignment evaluator* can be used to evaluate alignments using a reference alignments.

3.6 Repository

The Repository stores any reusable data for the alignment process.

4. IMPLEMENTATION

The semi-automatic approach (user input) was implemented as part of Falcon-AO. Implemented processes are discussed below.

4.1 User Input processes

Different processes have been implemented to process user specified input:

4.1.1 Parsing and Entity Retrieval Process

Parsing and Entity Retrieval Process is the first process that executes. The process first parses input ontologies and retrieves all the entities from the two ontologies and presents them on the GUI.

4.1.2 Model Creation and User Constraint Specification Process

This process implements user specified input by allowing user to select pairs of entities and a constraint for each pair. The process first creates an empty temporary ontology file and an empty *model* of it. Each selected entity is saved in the temporary ontology file then the next process is invoked.

4.1.3 Rule Implementation

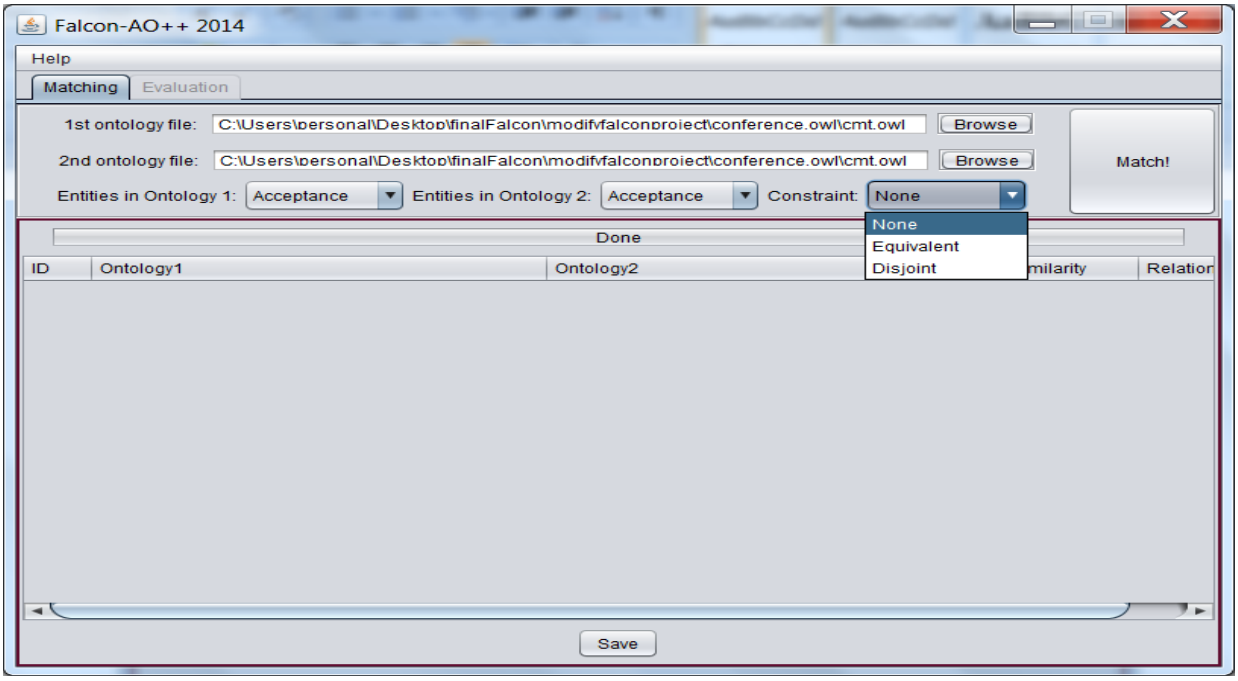
As soon as the user selects a constraint, the rule implementation process generates and implements a rule defining the relation between the entities in that same temporary file in stage 2. Rule implementation is done by *GenericRuleReasoner* provided by *jena.* If the selected constraint is *equivalent,* the rule defines the entities as *SameAs.* Otherwise if the selected constraint is *disjoint,* the rule defines the entities as *DisjointWith.*

4.1.4 Data Structures and Integration with Falcon-AO

After the rules have been implemented, the entities are further saved in data structures of the same type as Falcon-AO. This data structures are further integrated with the data structures of Falcon-AO for final output.

4.2 GUI of Falcon-AO++

Figure 2 shows the Graphical user interface of Falcon-AO++. The GUI of Falcon-AO was modified to allow the display of retrieved entities from the input ontologies. List of available constraints are also provided on the GUI.


1st ontology file: C:\Users\personal\Desktop\finalFalcon\modifyfalconproject\conference.owl\cmt.owl

2nd ontology file: C:\Users\personal\Desktop\finalFalcon\modifyfalconproject\conference.owl\cmt.owl

Entities in Ontology 1: Acceptance

Entities in Ontology 2: Acceptance

Constraint: None

Figure 2: GUI of Falcon-AO++ showing retrieved entities and constraints

5. EVALUATION AND RESULT

The Ontology Alignment Evaluation Initiative (OAEI) is a coordinated international initiative, which organizes the evaluation of the increasing number of ontology matching systems. To examine the impact of user input on the quality of alignment result, the OAEI conference track dataset consisting of different ontologies and reference alignments were used. The alignment results of the system are compared with alignments provided in the reference alignment. For the purpose of evaluation, the standard measures of *Precision*, *Recall, and F-measure* for evaluating ontology alignment systems were used.

$$Precison = \frac{Number\ of\ correct\ found\ alignments}{total\ number\ of\ found\ alignments}$$

$$Recall = \frac{Number\ of\ correct\ found\ alignments}{total\ number\ of\ existing\ alignments}$$

$$F - measure = \frac{2 * Precision * Recall}{Precision + Recall}$$

5.1 Experimental Setup

The experiments were performed on a PC with Intel (R) Core (TM) i3 CPU, 2.13GHz, 4 GB RAM, Windows 7, and Java NetBeans IDE 7.2

5.2 Result and Discussion

Table 1 gives the comparative alignment results of Falcon-AO and Falcon-AO++. The input ontologies are the ontologies needed to be matched (i.e., OAEI conference track ontologies), for each pair of input ontologies, the number of existing alignment in the reference alignment, the number of alignments found by each system, and the number of correct alignments found by each system are all given in the table. For Falcon-AO++, the number of inputs (**E** for equivalent and **D** for disjoint) provided by the user is specified.

Table 1: Alignment results of Falcon-AO and Falcon-AO++

			Falcon-AO		Falcon-AO++			
S/No	**Input Ontologies**	**No. of existing alignments in reference alignment**	**No. of found alignments**	**No. of correct found alignments**	**No. Of inputs**		**No. of found alignments**	**No. of correct found alignments**
					E	**D**		
1	Cmt conference	15	18	9	1	1	18	10
2	Cmt confOf	16	11	6	2	1	12	8
3	Cmt Edas	13	13	9	2	2	13	11
4	Cmt Ekaw	11	11	6	3	2	12	9
5	Cmt Iasted	4	8	4	0	3	5	4
6	Cmt Sigkdd	12	13	10	1	2	12	11

The values in Table 1 were used to obtain values for precision, recall, and f-measure for Falcon-AO and Falcon-AO++ using the previously specified formulas. The results are shown in Table 2.

Six pairs of input ontologies were used. For each pair, the values for precision, recall, and f-measure were computed. In the second result in Table 1, the user provided 2 inputs as equivalent and 1 input as disjoint. For these two inputs the values for precision, recall, and f-measure appreciated by 22.22%, 33.33% and 28.57%, respectively, as shown in Table 2. There is more increase in recall than precision because the number of equivalent inputs is higher than the number of disjoint inputs. In the last input, 1 input was provided as equivalent and 2 inputs as disjoint. With these there was 19.17%, 10.00% and 14.58% increase in precision, recall and f-measure respectively. In this case, there is more increase in precision than recall because more inputs were provided for disjoint. This shows that equivalent input increases recall and disjoint input increases precision. However, when same numbers of input pairs are provided for equivalent and disjoint, the increase is the same. This was observed from first and third inputs, where 11.11% increase was obtained for all three measures while 22.22% increase in was recorded for all three measures, respectively, for the first and third inputs. From the results in Table 1 and Table 2, the following observations are made:

1. The highest gain in precision, recall and f-measure was obtained in the fourth input pair as 37.49%, 49.99% and 43.48% respectively. This is because more user inputs were provided for this pair.

2. The least gain in precision, recall and f-measure was obtained in the first input as 11.11% for all the measures. This is because less user inputs were provided. From the results and observations, it is clear that even the minimum input provided by a user improves values for precision, recall, and f-Measure. This shows that user input has positive impact on alignment results. The comparative result in terms of average in precision, recall and f-measure is shown in Figure 3.

Table 2: Evaluation results of Falcon-AO and Falcon-AO++

		Falcon-AO			Falcon-AO++		
S/No	**Input Ontologies**	**Precision**	**Recall**	**F-measure**	**Precision**	**Recall**	**F-measure**
1	cmt conference	0.5	0.6	0.54545456	0.5555556	0.6666667	0.6060606
2	cmt confOf	0.54545456	0.375	0.44444445	0.6666667	0.5	0.5714286
3	cmt edas	0.6923077	0.6923077	0.6923077	0.84615386	0.84615386	0.84615386
4	cmt ekaw	0.54545456	0.54545456	0.54545456	0.75	0.8181818	0.7826087
5	cmt iasted	0.5	1.0	0.6666667	0.8	1.0	0.8888889
6	cmt sigkdd	0.7692308	0.83333333	0.8	0.9166667	0.9166667	0.9166667
Average		**0.5920746**	**0.674372133**	**0.61572133**	**0.75584048**	**0.79127818**	**0.76863456**

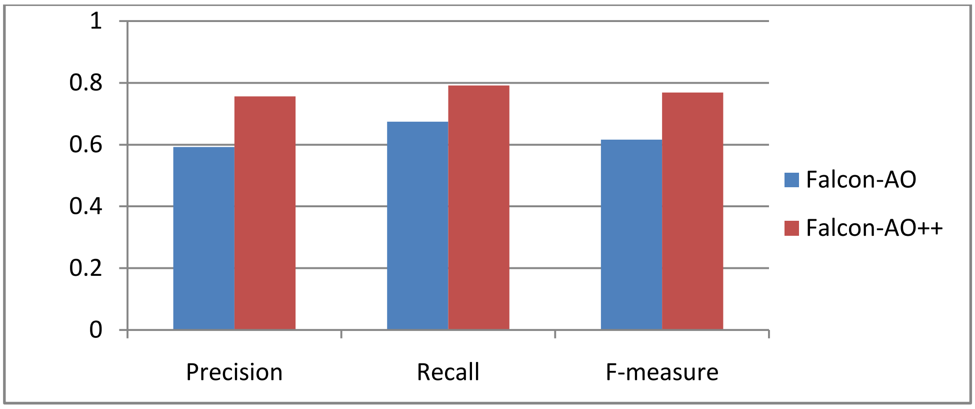


Figure 3: Comparative Result of Falcon-AO and Falcon-AO++

The comparative result in Figure 3 indicates clearly that Falcon-AO++ performs better than Falcon-AO. This shows that user assistance can significantly improve results of an alignment system.

6. CONCLUSION AND FUTURE WORK

In this paper, an ontology alignment system Falcon-AO++, based on Falcon-AO have been presented. The system supports the contribution of a domain expert. The two systems were evaluated using standard benchmark ontologies. The evaluation results have shown that user (domain expert) interaction has positive impact on alignment result.

Although user interaction improves results of an alignment system, providing input information is still a bottleneck since experts have to be manually involved especially when many inputs are passed. As a future improvement, it would be good to integrate more methods that would reduce the level of expert interaction. Presently, the system supports only one to one mapping and considers only equivalent relation. As another direction for future work, the system can be improved to support different kinds of mappings and also different relations. Also, *V-doc* can be extended to consider further neighbours rather than only one-step neighbours.

7. REFERENCES

[1] Shvaiko, P., & Euzenat, J. (2013). Ontology Matching: State art and Future Challenges. *IEEE* , pp.1-15.

[2] Jean-Mary, Y., Shironoshita, E., and Kabuka, M. (2009). Ontology Matching with Semantic Verification. *Journal of Web Semantics.* 7(3), 235–251

[3] Shen, G., Jin, L., Zhao, Z., Jia, Z., He, W., and Huang, Z. (2011). OMReasoner: Using Reasoner for Ontology Matching : results for OAEI 2011.

[4] Saruladha, K., Aghila, G., and Sathiya, B. (2011). A Comparative Analysis of Ontology and Schema Matching Systems. *International Journal of Computer Application*. *34*(8), 14-21.

[5] Gracia, J., and Mena, E. (2008). Ontology matching with CIDER: Evaluation report for the OAEI 2008. *In Proc. of 3rd Ontology Matching Workshop (OM'08), at 7th International Semantic Web Conference (ISWC'08), Karlsruhe, Germany.*

[6] Jian, N., Hu, W., Cheng, G., and Qu, Y. (2005). Falcon-AO: aligning ontologies with Falcon. *In Proceedings of K-CAP Workshop on Integrating Ontologies.* pp. 85–91.

[7] Granitzer, M., Sabol, V., Weng, K. O., Lukose, D., and Tochtermann, K. (2010). Ontology Alignment—A Survey with Focus on Visually Supported Semi-Automatic Techniques. *Future Internet* , *2*(3), 238-258

[8] Paulheim, H., Hertling, S., and and Ritze, D. (2013). Towards Evaluating Interactive Ontology Matching Tools. *The semantic web: semantics and big data, Lecture Notes in Computer Science LNSC*. pp.31-45.

[9] Sarasua, C., Simperl, E., and Noy, N. F. (2012). CROWDMAP: Crowdsourcing Ontology Alignment with Microtasks. *International Semantic Web Conference ISWC.* pp.525-541. Springer.

[10] Zhang, S., and Bodenreider, O. (2007). Lessons Learned from Cross-Validating Alignments between Large Anatomical Ontologies. *In Proceedings of 12th World Congress on Medical Informatics.* Brisbane, Australia.

[11] Vargas-vera, M., and Nagy, M. (2010). Towards Intelligent Ontology Alignment Systems for Question Answering: Challenges and Roadblocks. *Journal of Emerging technologies in web intelligence* , 2(3), 244-257

[12] Amrouch, S., and Mostefai, S. (2012). Ontology Interoperability Techniques, The State of the Art. *Journal of Information Organization* , 2(1), 20-27.

[13] Choi, N., Song, I., and Han, H. (2006). A Survey on Ontology Mapping. *SIGMOD Record*, 35 (3), 34-41.

[14] Hu, W., and Qu, Y. (2008). Falcon-AO: A practical ontology matching system. *Web Semantics: Science, Services and Agents on theWorldWideWeb*, *6*(3), 237-239.

[15] Hu, W., Qu, Y., and Cheng, G. (2008). Matching large ontologies: A divide-and-conquer approach. *Data and Knowledge Engineering* , *67*(1), 140-160.

[16] Qu, Y., Hu, W., and Cheng, G. (2006). Constructing virtual documents for ontology matching. *In Proceedings of the 15th International World Wide Web Conference.* pp. 23–31.

[17] Hu, W., Jian, N., Qu, Y., and Wang, Y. (2005). GMO: A Graph Matching for Ontologies. *In Proceedings of K-CAP Workshop on Integrating Ontologies*, pp.41–48.