

Evaluating Ontology Matchers Using Arbitrary Ontologies and Human Generated Heterogeneities

Nafisa Afrin Chowdhury and Dejing Dou

Department of Computer and Information Science
University of Oregon, Eugene, OR 97403, USA
{nafisa,dou}@cs.uoregon.edu

Abstract. Automatic ontology matching is a hard problem. To address this problem many ontology matchers have evolved in the past several years. Consequently the evaluation of ontology matchers has become crucial in order to help improve a matcher's performance. The evaluation frameworks used so far are limited to available pairs of ontologies in certain domains and require the reference alignments (i.e., gold standards) to be specified manually. In this paper we present a novel ontology matcher evaluation approach which can accept any OWL ontology as the source ontology. With little human efforts to specify the changes to the source ontology, our system can automatically construct the target ontology and generate the gold standard of correspondences. Compared to well-known evaluators (e.g., OAEI), our approach can provide more meaningful feedback besides traditional accuracy and completeness measures by indicating the performance of ontology matchers according to various types of heterogeneities.

Keywords: ontology matching, heterogeneity, evaluation.

1 Introduction

Automatic ontology matchers perform the job of finding correspondences between a pair of heterogeneous ontologies (i.e., the source and target ontologies) which are possibly in the same domain [11]. The set of correspondences or matchings is also called alignment [7]. In recent years many ontology matchers were developed based on different underlying ontology alignment algorithms, such as AFlood [8], Falcon [9], Aroma [4], Lily [15], Optima [5] etc. Now, as none of the matchers claim to be perfect as 100% accurate and complete, it is important to know their strengths and weaknesses. However, evaluating ontology matchers is a complicated task for three reasons [7]: a) different matchers use different algorithms, so some algorithms may work better for some type of heterogeneities than others, b) a pair of heterogeneous ontologies in any domain is not always available, and c) reference alignment to compare the outcomes of matchers is also rare and normally needs to be specified manually.

Ontology Alignment Evaluation Initiative (OAEI) [1] is a well known effort to evaluate ontology matchers and has significantly promoted the ontology matching research. However, the available pairs of ontologies along with the reference alignments provided by OAEI are limited to certain domains and cannot guarantee to have all types of heterogeneities in various domains. These openly available ontologies in OAEI have not been changed for the last several years [12]. So if a matcher performs well in OAEI evaluations, it is still uncertain how well it will do for another recently emerging pair of ontologies in the same or a different domain. Another challenge is to build the reference alignment, which is used as gold standard for evaluating ontology matchers. In most cases, the reference alignment is prepared by an ontology expert manually which is time consuming and very hard to a “non-matching-expert.” Although some evaluation approach [3] uses “inherent quality measures” in case of unavailable reference alignment, the effectiveness of these measures are still questionable.

With a pair of heterogeneous ontologies, the reference alignment, and the alignment generated by a matcher, a traditional evaluator provides the evaluation result. The result mostly consists of precision and recall values that indicate the “overall” accuracy and completeness of the matcher. Unfortunately these “overall” measures may not always be helpful to the matcher developers for further improvements, as these measures do not indicate what types of heterogeneities the matcher is good at or fails to deal with. In other words, it is difficult to infer from the measures what type of heterogeneities the matcher needs to improve.

Our evaluation approach is not restricted to any specific pair of ontologies or any set of domains. It also does not require the reference alignment. The user can upload any ontology as the source ontology and specify any type of heterogeneities. The system automatically constructs the target ontology and generates the reference alignment. Through our evaluation approach it is possible to apply different types of heterogeneities one at a time and measure the performance of the matcher or apply all of them together. In this way we can provide more meaningful evaluation results than the exiting well-known evaluation processes.

The rest of this paper is organized as follows: In section 2, we will do a short survey of related work on ontology matcher evaluation. In section 3 we will illustrate our framework, different types of heterogeneities available in our system and matcher evaluation metrics. In section 4, we will present our experiment of evaluating five different ontology matchers with eleven different types of heterogeneities in section 4. We will discuss our contributions, limitations and future work in section 5, and conclude the paper in section 6.

2 Related Work

OAEI [1] arranges a yearly competition for ontology matchers. The evaluation of OAEI is performed based on a set of ontology pairs and their reference alignments. The reference alignments are mostly created manually by ontology experts. OAEI has significantly promoted the ontology matching research. Since

the goal of the ontology matching problem is to find correspondences between two heterogeneous ontologies, it would be better that an evaluation system can provide testings with respect to certain types of heterogeneities. Our evaluation system can be compared to the OAEI competence benchmark, where the matchers are evaluated with regard to a set of tasks [13]. However, the results of OAEI do not directly specify what kind of heterogeneity the matcher needs to improve. Moreover, as we mentioned in the introduction, OAEI's evaluation is confined to a limited and static pairs of ontologies which have not been changed for the last several years [12]. So if a matcher performs well in OAEI evaluations, it is still uncertain how well it will do for another recently emerging pair of ontologies in the same or a different domain.

Rosoiu et al. [12] in their ontology matching benchmark paper identified OAEI's problem of using static ontologies and proposed a modular benchmark test generator. Like ours, their test generator accepts only one ontology and makes changes to generate an altered ontology. However, their alteration tasks are limited to removing and renaming classes and properties, and flattening hierarchies. The benchmark proposed by them is also completely program-specified (e.g., rename $x\%$ classes, flatten $y\%$ hierarchies etc.), which cannot handle certain types of heterogeneities that require domain knowledge.

Lambrix et al. [10] proposed the KitAMO framework for comparative evaluations of ontology alignment. Similar to other traditional evaluators, KitAMO requires a pair of ontologies and the reference alignment built by domain experts. Unfortunately, this framework is limited to certain matchers that can be added as plug-ins to the KitAMO framework. Our evaluation framework is not confined to any type of matchers and does not require any matchers to be directly added to the framework.

The Alignment API [6] proposed by Euzenut provides a widely accepted format for expressing ontology alignment. We have used their format while expressing the reference alignment generated by our evaluation system. The Alignment API also provides facilities for ontology matcher evaluations. However, like others, the API also requires a pair of ontologies and a well-defined reference alignment for the pair.

3 Our Framework

Our ontology matcher evaluation framework has three phases. In the first phase the user uploads an OWL ontology (as the source ontology) and introduces heterogeneities by changing the source ontology. The system keeps track of all changes made by the user and converts them into ontology matchings as the reference alignment (i.e., gold standard). Based on user generated heterogeneities, the system generates an OWL target ontology as well. The reference alignment is used to evaluate the ontology matchers. In the second phase the source ontology and generated target ontology are fed to an ontology matcher. Then the matcher's output alignment is fed to the system for evaluation. In the last phase the system performs comparison between the reference alignment and the

matcher’s alignment and, generates the evaluation results. The overall evaluation framework is shown in Figure 1.

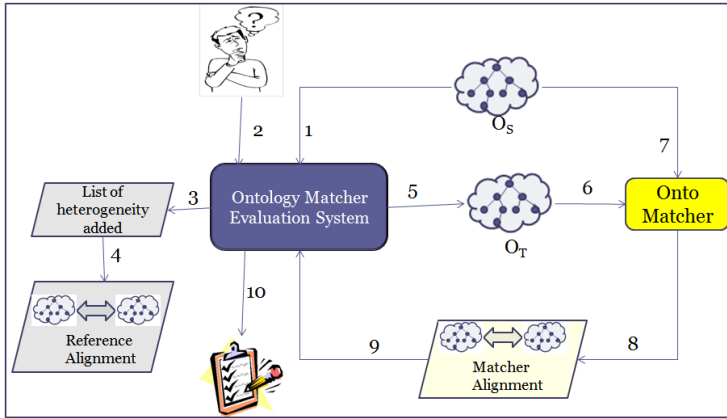


Fig. 1. Evaluation framework for ontology matchers. The user uploads the source ontology O_S (1) and input the heterogeneities between O_S and the target ontology O_T based on his knowledge and actions (2). The system generates the O_T (5) and the reference alignment as the gold standard of alignment between O_S and O_T (3, 4). Then the system feeds the O_S and O_T to an ontology matcher (6, 7), gets the matcher’s alignment result and feeds it to the system for evaluation (8, 9). Finally the system compares the matcher’s alignment with the reference alignment (10) and generates the evaluation results.

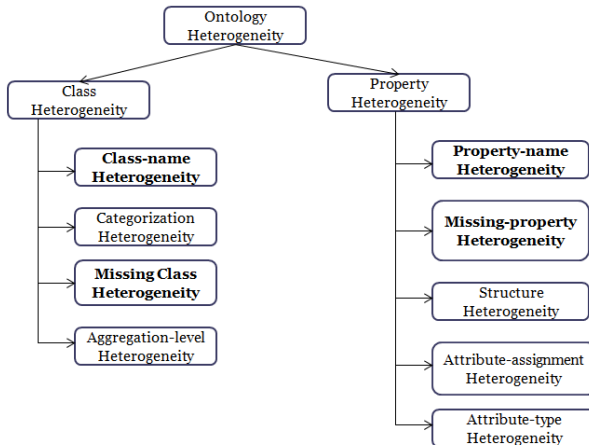


Fig. 2. Heterogeneity classification from the conceptualization point of view [14]. We have extended the classification with the four categories (in bold).

3.1 Types of Heterogeneities

Two ontologies are heterogeneous if they make different assumptions about their domain knowledge, as stated by Visser et al. in their ontology mismatches paper [14]. This paper presented a wide categorization of ontology heterogeneity. According to Visser et al., the categorization was presented from two different perspectives as it was hard to define all of them through one. Here we have adopted the conceptualization based categorization as it is more suitable with the current OWL ontologies and extended it to cover all other possible categories which are shown in Figure 2.

3.2 Class Heterogeneity

Heterogeneities that concern about classes only belong to the class heterogeneity.

- Class-name heterogeneity occurs when two classes from two ontologies defines the same concept but with different names (possibly synonyms). For example, one ontology defines ‘human’ and another defines the same concept as ‘person.’
- Categorization heterogeneity occurs when two ontologies distinguish a class into different subclasses [14]. An example is shown in Figure 3.

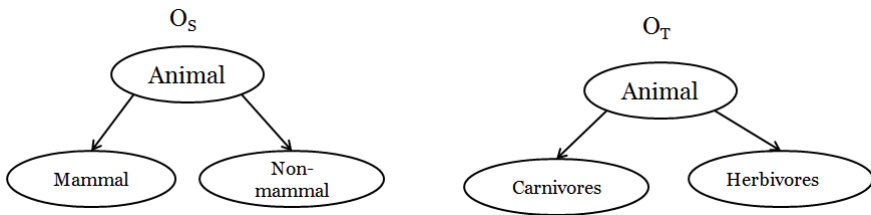


Fig. 3. Categorization heterogeneity: Ontology O_S and O_T classified from two different points of view

- Missing-class heterogeneity occurs when the source ontology defined a class, for which the target ontology has not defined anything. More precisely, the source ontology may contain a certain class or even a sub-tree which may not be considered implicitly or explicitly when designing the target ontology. For example (Figure 4), ‘virus’ does not belong to any of the subclasses in O_S . Note that this heterogeneity is different from the categorization heterogeneity, as for categorization heterogeneity all subclasses of one ontology can be defined by the categorization of the other ontology.
- Aggregation-level heterogeneity occurs when two ontologies define a class from different levels of abstraction [14].

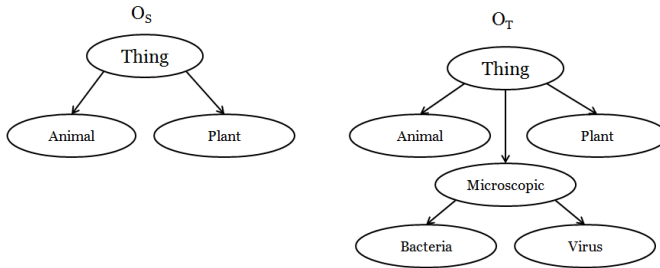


Fig. 4. Missing-class heterogeneity: O_S divides everything in animal and plant. But there are some creatures on earth which is neither an animal nor a plant, such as bacteria and virus in O_T .

3.3 Property Heterogeneity

Heterogeneities that concern properties belong to the property heterogeneity.

- Property-name heterogeneity occurs when one ontology has the same property as the other but with a different name (possibly synonym or different arrangement of words). For example ‘part-of’ vs. ‘is-part-of’, ‘executes’ vs. ‘performs’ etc.
- Missing-property heterogeneity occurs when one ontology defines a property and other does not. For example, two camera ontologies have classes ‘camera’ and ‘lens.’ One of the ontologies has a property <Domain: Camera, Property: compatible-with, Range: Lens> but the other does not have this property.
- Structure heterogeneity occurs when two ontologies have the same set of classes but differ in the way they are structured by means of properties [14]. For example if two ontologies have classes ‘house’ and ‘brick’ but they are related through different properties like ‘is-made-of’ and ‘has-component.’ Note this type of heterogeneity is different from property-name heterogeneity as in this type of heterogeneity two property names have different semantics.
- Attribute-assignment heterogeneity occurs when two ontologies differ in the way they assign a class to other through the property [14]. For example suppose there are three classes ‘vehicle’, ‘car’ and ‘color’ and, a property ‘has-color’ defined in both of the ontologies. But one of them defines ‘vehicle’ as the domain of ‘has-color’ and other defines ‘car’ as the domain.
- Attribute-type heterogeneity occurs when two ontologies distinguish the same class but differ in their instantiations [14]. For example both of the ontologies have defined the class ‘length’ and property ‘has-value’ but one assumes the range of ‘has-value’ should be ‘in-miles’ and the other assumes the range should be ‘in-kilometers.’

3.4 Evaluation System

The process of generating reference alignment based on user input is performed through a dynamic GUI. The user can first upload any OWL ontology as the

source ontology. Then our system creates a replica of the source ontology as the target ontology and shows both of the ontologies in a graphical view (Fig. 5). The user is allowed to introduce heterogeneities between these two ontologies by making “changes” on the target ontology. Our system¹ offers several different types of changes to perform on classes and properties of the target ontology. The changes implemented in the system is listed in Table 1 and Table 2.

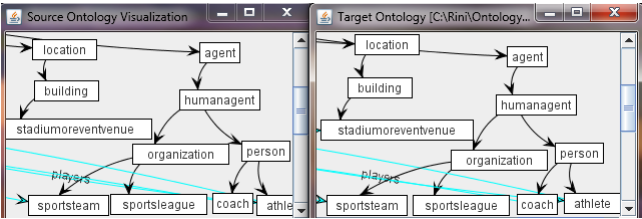


Fig. 5. Initial visualization of source and target ontologies for a sports domain. Classes are squared nodes, dark arrows are subclass relationships and light arrows are object properties. Object property arrows go from domain to range.

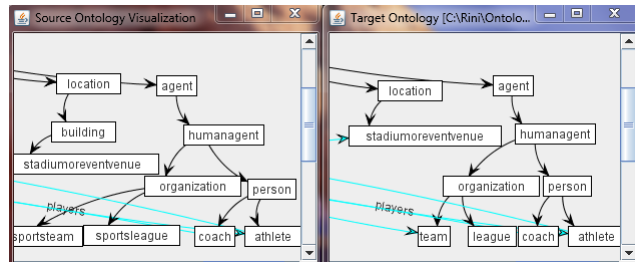


Fig. 6. Visualization of ontologies after generating some heterogeneities on the target ontology of Fig. 5. Here in the target ontology the ‘sportsteam’ and ‘sportsleague’ classes are renamed and the ‘building’ class has been deleted.

For introducing heterogeneities to the target ontology, the user can modify resources in the dynamic interface (Fig. 6). Each of these changes leads to a type of heterogeneity (described in the previous section) between the source and target ontologies, which is recorded by the system as a matching. The user can insert any of the heterogeneities described in the previous section. The list of heterogeneities and their corresponding user actions are shown in Table 1 and Table 2.

The user interface offers various changing options to mitigate user’s requirement for inserting heterogeneity, where user can see the changes dynamically. Figure 7 shows the user interfaces for the actions of changing the target ontology.

¹ Our evaluation system can be downloaded from <http://aimlab.cs.uoregon.edu/onlineservices.html>

Table 1. List of class heterogeneities, corresponding user actions and matching recognized by the evaluation system. \perp means “nothing.”



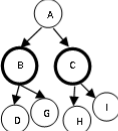
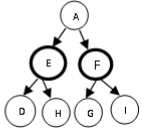


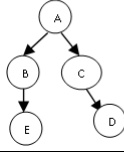
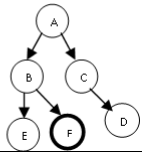
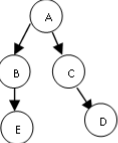
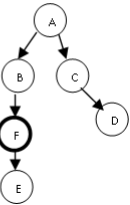
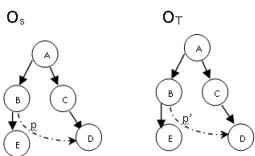
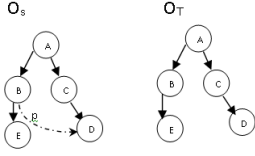
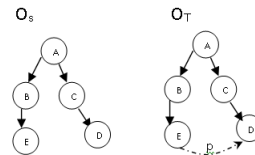
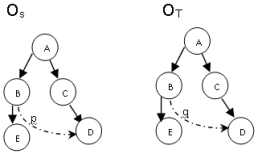
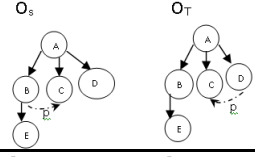
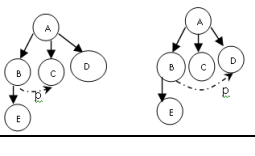
Types of Heterogeneity	Graphical Expression	User Action	Alignment
Class-name Heterogeneity	O_S  O_T 	Rename class A to A'	A matches to A'
Categorization Heterogeneity	O_S  O_T 	Delete class B and C but keep subclasses. Add E and F. Change parents of D, G, H and I to E or F according to definition of E and F.	B matches to \perp . C matches to \perp . \perp matches to E. \perp matches to F.
Missing-class Heterogeneity (O_S has an additional class)	O_S  O_T 	Delete class B	B matches to \perp
Missing-class Heterogeneity (O_S has a missing class)	O_S  O_T 	Add class F as a subclass of B and sibling of E	\perp matches to F
Aggregation-level Heterogeneity	O_S  O_T 	Add class F as a subclass of B. Change parent of E to F.	B matches to F

Table 2. List of property heterogeneities, corresponding user actions and matching recognized by the evaluation system. \perp means “nothing.”

Types of Heterogeneity	Graphical Expression	User Action	Alignment
Property-name Heterogeneity		Rename property p to p'	p matches to p'
Missing-property Heterogeneity (additional property in O_S)		Delete property p	p matches to \perp .
Missingproperty Heterogeneity (O_S has a missing property)		Add a property p specifying the domain and range of p	\perp matches to p.
Structure Heterogeneity		Delete property p. Add property q such that p and q are semantically different.	p matches to \perp . \perp matches to q.
Attribute-assignment Heterogeneity		Change the domain of property p	B matches to D.
Attribute-type Heterogeneity		Change the range of the property p	C matches to D.

3.5 Ontology Matcher Evaluation

When the user is done with inserting heterogeneities by changing the target ontology, our system converts them to ontology matchings and generates the target ontology in OWL. The system considers the set of generated matchings as the gold standard alignment and uses it as the reference alignment while evaluating matchers. In the next step, both the source and target ontologies are fed to an ontology matcher to obtain matcher's alignment. In the evaluation step our system compares the matcher's alignment with the reference alignment with the help of ontology API[6] and generates the evaluation results. The evaluation result includes precision, recall and f-measure which denote the accuracy, completeness and usability of the matcher.

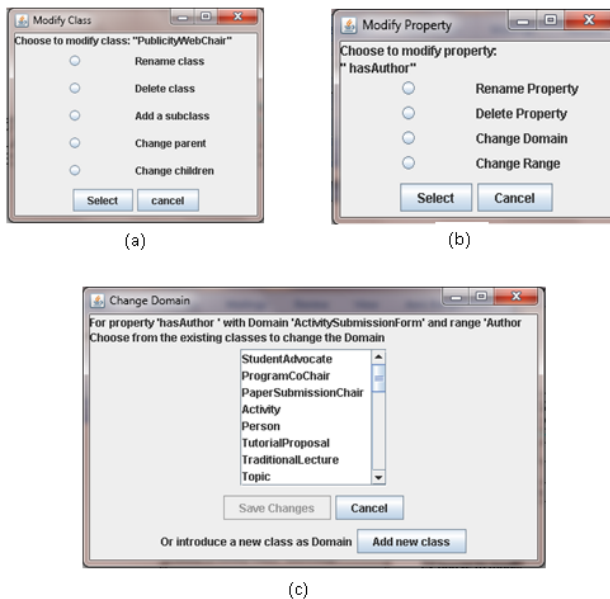


Fig. 7. UI options for (a) modifying a class, (b) modifying a property, (c) change the domain of a property

4 Experiments

As our system allows any OWL ontology to be used for evaluation, we have tested matchers' performances in two different domains: conference and camera.

4.1 Conference Ontologies

In order to justify our system and compare with the OAEI efforts, we have experimented with the conference ontologies used during the OAEI 2012 [1]. We have

chosen the conference ontologies because the semantics of conference domain is easier for us to understand than other domains like anatomy or gene. Another important issue was scalability. As our system relies on user intervention, a very large ontology would increase the difficulty of user intervention as well as the evaluation process. The ontologies of the conference domain are of moderate size (e.g. 14-140 classes). We have tested the biggest ontology in the conference domain called “lasted” with 140 classes and the our system takes 788 milliseconds to load and draw the source ontology and initial target ontology. For performing matcher evaluation experiments we have chosen Micro.owl, ConfOf.owl and SigKDD.owl because of their moderate size and commonality in concepts.

In order to evaluate the ontology matchers, we were interested to measure their strengths and weaknesses at the most granular level. Therefore, we performed evaluation regarding each type of heterogeneities one after another rather than mixing the heterogeneities like the ontology matching benchmark in [12]. While changing the target ontology, we have looked at other conference ontologies for gathering candidate heterogeneity ideas. The following are the steps of evaluation in a nut shell:

1. For all 11 types of heterogeneities $H_1..H_{11}$ with index i (see table 1 and 2)
 - (a) Load an ontology as source ontology O_S
 - (b) our system draws the initial target ontology O_{Ti} same as O_S
 - (c) Apply H_i following the instructions in table 1 and 2
 - (d) Get the modified O_{Ti} and Reference alignment A_{Ri}
 - (e) For n ontology matchers $M_1..M_n$ with index j
 - i. Apply O_S and O_{Ti} to M_j
 - ii. Get matcher’s alignment A_{Mj}
 - iii. Apply A_{Ri} and A_{Mj} to our system
 - iv. our system generates the evaluation result
 - (f) End of For
2. End of For

We have experimented with five ontology matchers while doing the evaluation: Anchor Flood [8], Falcon AO [9], AROMA [4], Lily [15] and Optima [5]. The reason for choosing these five matchers is that all of them took part in the OAEI competition in different years and experienced the conference ontologies. According to the results of OAEI, Anchor Flood and Falcon are expected to perform better than the other three.

Experimental Results: Table 3 shows the detail evaluation results for different ontology matchers with respect to different types of heterogeneities. We can see that the performance of a matcher varies while dealing with different types of heterogeneities. Some matchers achieved very high evaluation scores on certain types of heterogeneities but their performance were not satisfactory in other types. In order to analyze the matchers’ performance in detail, we have plotted their precision and recall individually (Fig. 8). This figure clearly shows that Anchor Flood and Falcon received the best recall value. The four

Table 3. Evaluation results for all matchers with respect to different types of heterogeneities. Here we have used Micro.owl as the source ontology and looked at ConfOf.owl and SigKDD.owl ontologies before inserting heterogeneities into the target ontology. We inserted five class-type heterogeneities and three property-type heterogeneities at a time.

Types of Heterogeneity	Measure	AFlood	Falcon	Aroma	Lily	Optima
class-name and property-name	Precision	1.0	1.0	1.0	0.982	1.0
	Recall	1.0	1.0	0.947	0.965	0.421
	F1	1.0	1.0	0.973	0.973	0.592
Categorization	Precision	1.0	1.0	1.0	1.0	1.0
	Recall	0.983	0.983	0.983	0.983	0.465
	F1	0.991	0.991	0.991	0.991	0.635
Missing class (missing in Os)	Precision	1.0	1.0	1.0	0.892	1.0
	Recall	1.0	1.0	1.0	0.877	0.474
	F1	1.0	1.0	1.0	0.885	0.643
Missing class (additional in Os)	Precision	1.0	1.0	1.0	0.958	1.0
	Recall	1.0	1.0	1.0	0.939	0.408
	F1	1.0	1.0	1.0	0.948	0.580
Aggregation level	Precision	1.0	1.0	1.0	1.0	1.0
	Recall	0.983	0.983	0.965	0.983	0.465
	F1	0.991	0.991	0.982	0.991	0.635
Missing property (missing in Os)	Precision	1.0	1.0	1.0	1.0	1.0
	Recall	1.0	1.0	1.0	0.982	0.474
	F1	1.0	1.0	1.0	0.991	0.643
Missing property (additional in Os)	Precision	1.0	1.0	1.0	1.0	1.0
	Recall	1.0	1.0	1.0	0.981	0.519
	F1	1.0	1.0	1.0	0.990	0.683
Structure	Precision	1.0	0.947	1.0	0.981	1.0
	Recall	1.0	1.0	1.0	0.981	0.500
	F1	1.0	0.973	1.0	0.981	0.667
Attribute-assignment	Precision	1.0	1.0	1.0	1.0	1.0
	Recall	0.950	0.950	0.950	0.95	0.45
	F1	0.974	0.974	0.974	0.974	0.621
Attribute-type	Precision	1.0	1.0	1.0	1.0	1.0
	Recall	0.950	0.950	0.950	0.95	0.45
	F1	0.974	0.974	0.974	0.974	0.621

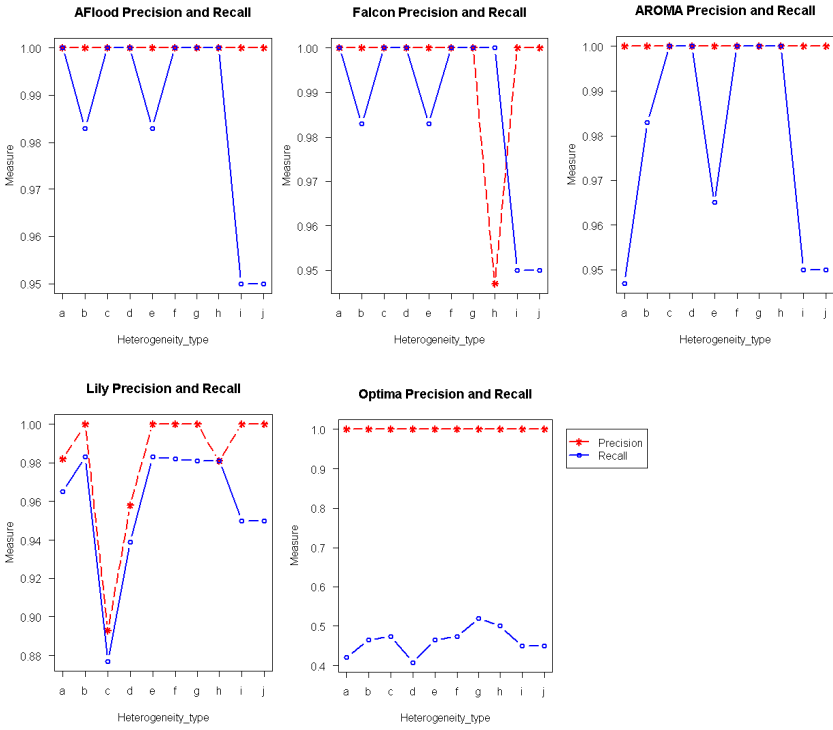


Fig. 8. Precision and recall value plotted for the individual matchers. In this Figure a means the class-name and property-name heterogeneity, b means the categorization heterogeneity, c means the missing class heterogeneity (missing in O_S), d means the missing class heterogeneity (additional in O_S), e means the aggregation level heterogeneity, f means the missing property (missing in O_S) heterogeneity, g means the missing property (additional in O_S) heterogeneity, h means the structure heterogeneity, i means the attribute-assignment heterogeneity, and j means the attribute-type heterogeneity.

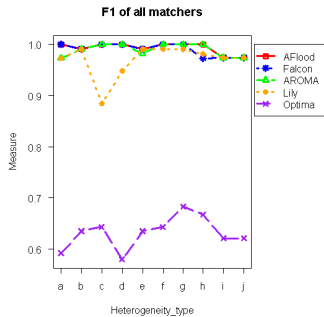


Fig. 9. Comparing all five matchers based on their F1 value with respect to different types of heterogeneities

types of heterogeneity where they failed to capture all correct matchings were: b (the categorization heterogeneity), e (the aggregation-level heterogeneity), i (the attribute-assignment heterogeneity) and j (the attribute-type heterogeneity). Falcon had the lowest precision for the structure heterogeneity (type h) among all the matchers. Surprisingly, AROMA performed much better than their OAEI 2012 results [1]. In fact, AROMA received the best precision along with Anchor Flood and Optima. But in terms of recall AROMA showed unexpected low recall for the class-name and property-name heterogeneities (in Fig. 8 type a), which can be assumed to be the easiest type of matching to detect. Like AROMA, Lily and Optima also performed worse than the baseline 1 in OAEI 2012 [1]. In our experiment, their performances were also not as good as the others. Optima showed a very large difference between precision and recall. Although its precision was quite satisfactory, it showed very low recall values for all types of heterogeneities.

For comparing the matchers with each other we have plotted their F1 values against different types of heterogeneities in Figure 9. As expected, Anchor Flood, Falcon and AROMA were found having F1 values close to 1.0 for all different types of heterogeneities except the Missing-class heterogeneity (missing in O_S). Lily's F1 values were surprisingly very close to Anchor Flood, Falcon and AROMA. Unfortunately, Optima could not defeat any other matchers in any type of heterogeneity.

4.2 Camera Ontology

To test our system in an arbitrary OWL ontology which has not been used in the OAEI library, we chose a camera ontology [2] as the source ontology. We have conducted the same experiment as we did in the conference ontologies: basically we generated all 11 types of heterogeneities by changing the target (camera) ontology. Then we evaluated the same five ontology matchers in our system. Optima was unfortunately unable to save the resulted alignment from the camera ontologies. Because of the space limit, we only report the results of the camera ontologies compared with the results from the conference ontologies. The comparison of these two experiments have been shown in Figure 10. Though we generated the same number of heterogeneities in both of the experiments in order to perform a fair comparison, some matchers' performances were inconsistent across different domains. For example, for the camera ontologies both Anchor Flood and Aroma dropped their F1 values greatly in class-name and property-name heterogeneity test. Anchor Flood, Falcon and Aroma's overall performance were better in the conference ontologies rather than in the camera ontologies.

Now, instead of testing the matchers with respect to each type of heterogeneity, we could apply all types of heterogeneities together and measure the performances (like the benchmark generator [12]). Figure 11 depicts the matchers' performances when all types of heterogeneities were applied together. It is clear from the figure that recall values were always lower than the precision and Optima obtained lowest recall than others. However, it is almost impossible from

this figure to infer what type of heterogeneities a matcher (e.g., Optima) had difficulties to deal with. Figure 8 provides a more detailed evaluation feedback than Figure 11 by precisely indicating a matcher’s strengths and weaknesses.

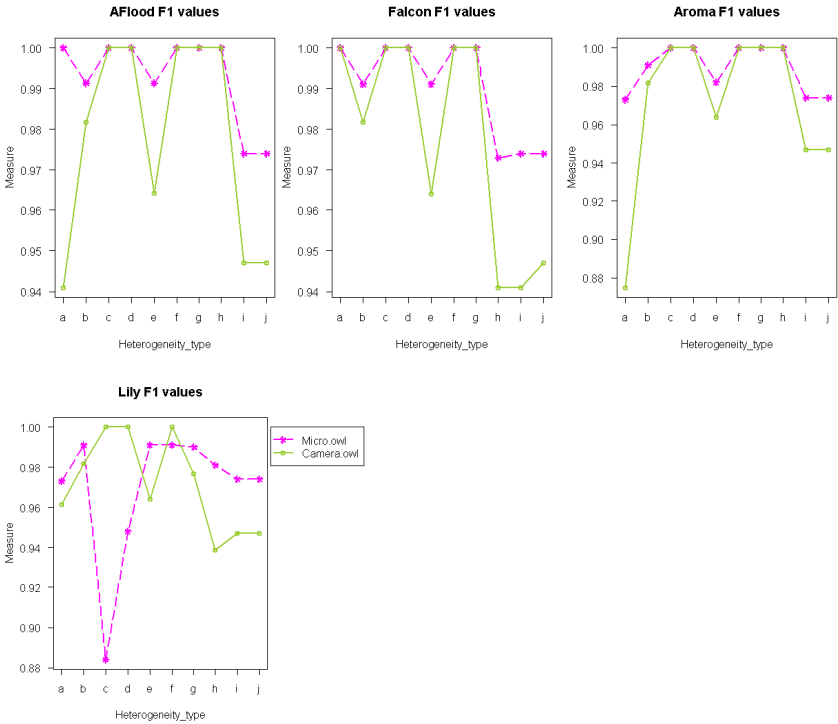


Fig. 10. Comparing matchers’ F1 values in two different domains: Conference and Camera. The comparison could not be done for Optima as Optima was unfortunately unable to run the camera ontologies.

5 Discussion

We have studied the reference alignments of conference ontology pairs provided by the OAEI documentation [1] for understanding how many types of heterogeneities these ontology pairs provide. We found that most of the pairs provide naming heterogeneities (e.g., class-name or property-name heterogeneities) and only a few structure heterogeneities. Our evaluation system provides other eight types of heterogeneities like categorization, aggregation, attribute-assignment, attribute-type, missing-class and missing-property heterogeneities that can be generated by human actions. We have performed evaluation of five ontology matchers for all eleven types of heterogeneities.

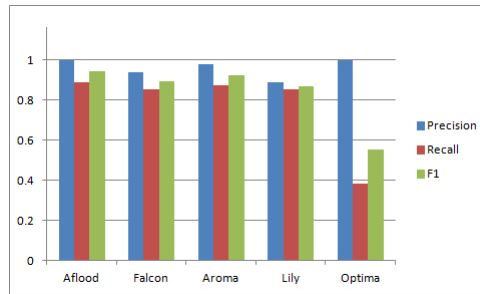


Fig. 11. Comparing all five matchers after applying all types of heterogeneities together. Micro.owl was used as the source ontology and all the user actions used to introduce heterogeneities in Fig. 8 were combinedly applied here.

In the experimental results, it is noticeable that for both the attribute-assignment and attribute-type heterogeneities the highest recall value was 0.95. This means when the domain or range of a property was changed, none of the matchers was able to detect all the correct matchings. In fact, if the user changes the domain (or range) of a property while keeping the definitions of the property, the old domain (or range) and the new domain (or range) same, there should be a correspondence between the old domain (or range) and new domain (or range). Unfortunately, the matchers could not find the matchings generated by these type of heterogeneities. There could be two reasons behind this: 1) the matcher developers might have not considered these two types of heterogeneities and so the matchers failed to detect the matchings because of these two, 2) these two types are hard to detect.

If we evaluate the matchers after mixing all types of heterogeneities together (like we did in Figure 11), it would be hard to justify their weak (or strong) points. In order to identify the strengths and weaknesses of a matcher, it is necessary to evaluate them against different types of heterogeneities. Thus the matcher developer will get more detailed feedback to help their future improvements. Now, we can summarize key points of our evaluation process:

- Our evaluation system accepts any OWL ontology of an arbitrary domain.
- Unlike traditional ontology matcher evaluators our evaluator can measure the performance of a matcher against each type of heterogeneity.
- Our evaluation process requires only one source ontology rather than a pair of heterogeneous ontologies required by traditional evaluators.
- After uploading the source ontology the user can introduce different types of heterogeneities as much as he wants and our system can generate the heterogeneous target ontology based on user input.
- In order to compare matcher's alignment with the gold standard, traditional evaluators require a reference alignment. In our evaluation process, the system keeps track of the changes the user made on the target ontology and converts these changes to the reference alignment.

- Both the generated heterogeneous target ontology and the reference alignment are reusable, so our system can also be used for creating a pair of heterogeneous ontologies with minimal effort, as the user does not need to start from the beginning.

As the very first prototype of the evaluation system, we still have some limitations that we expect to address in the future.

- We need to extend our ontology visualization with equivalent classes, datatype properties, multiple superclasses and property hierarchies.
- Our ontology visualization is not yet capable of viewing very large ontologies.
- Our system deals with equivalence matchings only. We need to incorporate other matching types like subsumption, disjointness and ontology mapping in the future.
- The evaluation of instance matching is not yet available in our system.

6 Conclusion

To address automatic ontology matching problem, many ontology matchers have been developed in the last decade. However, no matcher can claim to be the perfect and complete. Therefore, it is necessary to have a comprehensive evaluation process for matchers' future improvement. Although several evaluation efforts have been acknowledged, almost all of them require a pair of heterogeneous ontologies and a reference alignment. A pair of heterogeneous ontologies in any domain is rarely available [7]. Even if a pair of ontologies can be found, building a reference alignment for it, is hard and time consuming [11]. Our system made this evaluation process easier by using only one ontology and the user inputs (for generating heterogeneities). Our system infers user's expected matchings by tracking user actions (i.e., changes to the target ontology) and automatically builds the reference alignment to evaluate the matchers. Almost all traditional evaluation systems provide an overall score (precision and recall) and do not precisely indicate the strengths or weaknesses of a matcher. Our evaluation system helps users to test a matcher for different types of heterogeneities individually. Users can also introduce different combinations of heterogeneities and evaluate a matcher. To validate our system, we have carried out an experiment to evaluate five different matchers and measured their performances for eleven types of heterogeneities. We found that matchers' recall values are much lower than the precision, which means matchers need to address more varieties of heterogeneities rather than improving the accuracy only. Our system can provide more detailed results indicating matchers' strengths and weaknesses in dealing with different types of heterogeneities which will help matcher developers in more granular level. We hope in the future we will extend our evaluation system and will be able to provide more detailed evaluation feedback to the matcher developers.

Acknowledgement. This work was supported by the National Science Foundation grant IIS-1118050.

References

1. <http://oaei.ontologymatching.org/>
2. <http://protege.cim3.net/file/pub/ontologies/camera/camera.owl>
3. Cruz, I.F., Antonelli, F.P., Stroe, C.: Agreementmaker: Efficient matching for large real-world schemas and ontologies. *PVLDB* 2(2), 1586–1589 (2009)
4. David, J., Guillet, F., Briand, H.: Matching directories and OWL ontologies with AROMA. In: *CIKM*, pp. 830–831 (2006)
5. Doshi, P., Kolli, R., Thomas, C.: Inexact matching of ontology graphs using expectation-maximization. *J. Web Sem.* 7(2), 90–106 (2009)
6. Euzenat, J.: An API for Ontology Alignment. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) *ISWC 2004. LNCS*, vol. 3298, pp. 698–712. Springer, Heidelberg (2004)
7. Euzenat, J., Shvaiko, P.: *Ontology Matching*. Springer (2007)
8. Hanif, M.S., Aono, M.: An efficient and scalable algorithm for segmented alignment of ontologies of arbitrary size. *J. Web Sem.* 7(4), 344–356 (2009)
9. Jian, N., Hu, W., Cheng, G., Qu, Y.: FalconAO: Aligning Ontologies with Falcon. In: *K-CAP Integrating Ontologies Workshop*, pp. 85–91 (2005)
10. Lambrix, P., Tan, H.: A Tool for Evaluating Ontology Alignment Strategies. In: Spaccapietra, S., Atzeni, P., Fages, F., Hacid, M.-S., Kifer, M., Mylopoulos, J., Pernici, B., Shvaiko, P., Trujillo, J., Zaihrayeu, I. (eds.) *Journal on Data Semantics VIII. LNCS*, vol. 4380, pp. 182–202. Springer, Heidelberg (2007)
11. Noy, N.F., Musen, M.A.: PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. In: *AAAI/IAAI*, pp. 450–455 (2000)
12. Rosoiu, M.E., dos Santos, C.T., Euzenat, J.: Ontology matching benchmarks: generation and evaluation. In: *The Sixth International Workshop on Ontology Matching* (2011)
13. Shvaiko, P., Euzenat, J.: Ontology Matching: State of the Art and Future Challenges. *IEEE Transactions on Knowledge and Data Engineering* (2011) (accepted)
14. Visser, P.R.S., Jones, D.M., Bench-capon, T.J.M., Shave, M.J.R.: An Analysis of Ontology Mismatches; Heterogeneity Versus Interoperability. In: *AAAI 1997 Spring Symposium on Ontological Engineering*, pp. 164–172 (1997)
15. Wang, P., Xu, B.: Lily: Ontology Alignment Results for OAEI 2009. In: *The Fourth International Workshop on Ontology Matching*, pp. 186–192 (2009)