

ASMOV: Ontology Alignment with Semantic Validation

Yves R. Jean-Mary, MSEC¹

¹INFOTECH Soft, 9200 South Dadeland Blvd, Suite 620, Miami, Florida, USA 33156

²University of Miami, Coral Gables, Florida, USA 33124

reggie@infotechsoft.com

Mansur R. Kabuka, PhD^{1,2}

¹INFOTECH Soft, 9200 South Dadeland Blvd, Suite 620, Miami, Florida, USA 33156

²University of Miami, Coral Gables, Florida, USA 33124

kabuka@infotechsoft.com

ABSTRACT

Numerous ontology alignment algorithms have appeared in the literature in recent years, but only a few make use of the semantics enclosed within the ontologies in order to improve the accuracy. In this paper, we present ASMOV (Automated Semantic Mapping of Ontologies with Validation), a novel algorithm that expands upon the ideas presented by previous systems, while incorporating a semantic validation process. This process acts as a reasoner over the resulting alignment, ensuring that no inconsistencies have been introduced by the system, which increases the accuracy of the system. An implementation of ASMOV is compared to other systems using the benchmark tests of the Ontology Alignment Evaluation Initiative, a consensus for evaluation of ontology alignment systems.

1. INTRODUCTION

In recent years, ontology alignment (or ontology mapping) has become popular in solving interoperability issues across heterogenous systems in the semantic web. Though many techniques have emerged from the literature [7][18], the distinction between them is accentuated by the manner in which they exploit the features within an ontology. Other than the relations between entities (concepts and properties), an ontology may also contain textual features: human-readable descriptions of the entities. The accuracy of current alignment techniques fluctuates depending on the amount of knowledge within the ontologies that is exploited. Early research focused only on the similarity within the taxonomy structure of ontologies and string distances between the labels of their concepts [2]. The comparison of the taxonomy structure combined with the calculation of similarity between labels is insufficient for several reasons, most notably because an ontology language only provides tools to represent the knowledge and does not dictate how this knowledge representation is to be achieved. Thus, two ontologies built by two people for the same domain might be entirely different in structure. The early practice will not be able to create a mapping between ontologies properly since its primary assumption of similar taxonomy structures may be false. Other systems, such as FCA-Merge [20], improve upon this technique by exploiting both the parent-child relationships between entities and the taxonomy structure of the ontologies and perform best when ontologies share the same individuals (instances). However, not all

ontologies include individuals, and thus approaches similar to FCA-Merge and T-tree [6] perform poorly in this case. The PROMPT suite which includes an interactive ontology merging tool [17] and a graph-based mapping (Anchor-PROMPT) [16] introduces two additional features in the ontology alignment process. It allows users to provide feedback for partial mapping, and it investigates the internal structure of concepts and properties. The result produced by PROMPT is not a mapping, but instead is a set of suggestions by which concepts and properties can be merged or aligned. Therefore, the process is not automated as the user is ultimately responsible to create the mapping file. S-Match [8][9], which uses WordNet [22] for semantic matching between textual descriptions of ontology entities, also runs a “decider” for propositional satisfiability on the preliminary mappings to determine further mappings. Additionally, S-Match uses the relationship between concepts within an ontology to discover similarities. In order to discover a mapping, GLUE [3] exploits information within the concept instances and taxonomy structure of ontologies through the use of multiple learners. RiMOM [21] employs features found among other methods within its technique. It accepts user inputs to improve matching similar to PROMPT and uses WordNet to discover semantic similarities in textual descriptions like S-Match. RiMOM also exploits instances, entity name, entity description, taxonomy structure, and constraints before using Bayesian decision theory in order to generate a mapping between ontologies. Other systems such as Falcon-AO [10] and COMA [15] cannot be ignored as they have performed well in ontology alignment contests. In both of these systems, the alignment is derived by graph-based matching techniques. OLA [4], where alignment is also produced by a graph-based analysis, uses weights in order to compute the similarity measures between concepts and properties.

In this paper, we present ASMOV, an algorithm that automates the ontology alignment process while optionally accepting feedback from a user. ASMOV builds upon the ideas used by previous techniques, using automatically-adjusting weights based on the features of the ontologies and evaluating the universality of features, or all of the available knowledge, within the alignment process. Although similar to OLA, ASMOV computes similarity measures by analyzing the entities in the manner in which they are modeled in the ontology, whereas OLA computes its similarity measures based on a graph created with entities and inter-entities relationships. And, unlike OLA, the iterative alignments produced by ASMOV are validated by a number of rules and a mapping validation process. These differences allow ASMOV to outperform OLA and other systems in many benchmark tests as is shown in the results section.

The rest of the paper is organized as follows: Section 2 and 3 present the techniques and calculations employed to achieve the alignment between two ontologies. The algorithm is evaluated against standard ontology matching problems and the

experimental results are presented in Section 4. Finally, we discuss future work before concluding the paper.

2. SIMILARITY CALCULATIONS

The four key features of an ontology that are exploited by ASMOV to match pair of entities include: the lexical information, the internal structure, the external structure and the individuals. Four partial similarity calculations are derived from these features:

- sim_L measures lexical similarity including identifiers, labels, and descriptions;
- sim_I measures internal structure similarity, that is, similarity between properties of classes and between the domain and range of properties;
- sim_E measures external structure similarity, including relations between the parents and the children of classes or properties; and
- sim_N measures individual similarity, that is, the similarity between the sets of class members.

These four partial similarity measures are combined in a weighted-sum calculation resulting in a single similarity confidence. Consider $F=\{L,I,E,N\}$ to be the set of features and e_1 and e_2 to be entities belonging to two ontologies (respectively \mathcal{O}_1 and \mathcal{O}_2), the similarity confidence is calculated as follows:

$$sim(e_1, e_2) = \frac{\sum_{i \in F} w_i \times sim_i(e_1, e_2)}{\sum_{i \in F} w_i} \quad (1)$$

where w_i are weights assigned to each of the features in the calculation. Using fixed weights presents a problem, as noted in [1]: if a given feature is missing (e.g., if an ontology does not contain individuals), the comparison becomes non-uniform. To compensate for this, the denominator in equation (1), which is the sum of the weights of the features present in the calculation, ensures that the ratio between weights is maintained and that their sum adds up to 1.

2.1 Lexical similarity (Sim_L)

The lexical feature space is the human-readable information stored in an ontology. In RDF, this information is enclosed in the identifier, label, and comment describing an entity. In the pre-processing phase of our algorithm, all identifiers and labels are tokenized. Uppercase and non-alphabet characters indicate where the text is to be split. This ensures that the terms match the expected format of lexical reference systems (thesaurus). WordNet, which is a thesaurus of English words, is used by default to validate the tokens and retrieve the semantics of the words. Each of the words that are discovered is subsequently labeled with its part-of-speech and stored with all of its associated semantic information. In some instances a word can belong to multiple parts-of-speech. When both words being compared belong to multiple parts-of-speech, a similarity value is calculated separately for each part-of-speech the words have in common. The highest resulting similarity measure is used to establish the distance between the words. The semantic information retrieved from the lexical reference system is composed of sets of synonyms, antonyms, and hypernyms. A synonym set, which is called a synset in WordNet, is a set of synonyms tied to a given definition of a word. A word has a synset for each one of its definitions. If one of the synonyms of the source word is the same as one of the synonyms of the target word, the words are

determined to be semantically equal. On the other hand, if the synset of the source word contains an antonym to any of the synonyms from the target word, the words are determined to be semantically orthogonal. When the words are neither synonyms nor antonyms, the hypernym of each of the definitions is traversed recursively to find the shortest path to a common definition. ASMOV uses the semantic distance equation proposed by Lin [14] to compute a measure of textual similarity:

$$Lin(s_s, s_t) = \frac{2 \times \log P(s)}{\log P(s_s) + \log P(s_t)} \quad (2)$$

where $P(s)$ is the probability of randomly selecting a word of WordNet that belongs to the synset S or any hyponyms of that synset. S_s and S_t are respective synsets of the source and target words. $Lin(s_s, s_t)$ will vary between 0 and 1. The higher the value, the shorter the path is from the source synset to the target synset. For each hypernym found, it is also necessary to look for its antonym; the algorithm infers that if the parents of two words are antonyms, the words must also be antonyms. If the tokenization process does not produce any meaningful words, the text is used in its original form and the Levenshtein distance is used to calculate the similarity.

Since the comments of an entity are typically in natural language, it is pre-processed differently from the identifiers and labels. First, all non-alphanumeric characters are removed from the text; then the text is parsed into a collection of words. The similarity measure is given by the intersection of the collection of words:

$$Sim_c(t_s, t_t) = 1 - \frac{C(t_s) \cap C(t_t)}{size(C(t_s)) + size(C(t_t))} \quad (3)$$

t_s and t_t are respectively the comment for a source and target entity. The function C returns the collections of words and $size$ returns the number of words in a collection. The lexical similarity is then calculated using the equation (1) where the features are the identifiers, labels and comments and their respective weights are w_{id} , w_{label} , $w_{comment}$.

2.2 External similarity (Sim_E)

The external similarity is computed by combining the similarities between the parents and children of the entity being compared. As entities may contain multiple parents and children, the similarity calculation is normalized in order to restrict the results between 0 and 1. For example, each parent of the source entity is paired with the closest parent of the target entity; all parent-pair similarity measures are summed and then normalized by dividing by the total number of parent entities for both the source and target entities. To combine the two similarity measures (parent and child), equation (1) is used once again.

2.3 Internal Similarity (Sim_I)

Two types of entities exist in an ontology: concepts and properties. Since a concept's internal constructs are different than a property's internal constructs, their internal similarity measure is computed differently

Concept internal similarity: the properties that define the concepts contribute to the similarity measure. This similarity measure is a combination of the similarity between properties and the local restrictions (cardinality and value restrictions). As the

number of properties associated to a concept may exceed one, the similarity measure must be normalized in a similar fashion as the external similarity.

Property internal similarity: two categories of properties exist in an ontology: data type and object properties. The major difference between them is their range. In data type properties, the range is usually an XML data type, whereas in the object properties, the range is a concept. Since an ontology developer has the liberty of modeling an object property as a data type property or vice versa, ASMOV does restrict the alignment between properties; properties of different categories may be aligned. However, before allowing the latter, the object property is investigated in order to check if it could be considered as a data type property. The calculation of internal similarity of properties entails comparing the type attribute, domain, and range. The type attributes are pre-defined constructs, therefore, the similarity between two types can only be 1 if the types are the same and 0 otherwise. The domain of a property is a concept or a collection of concepts; therefore, the similarity between the best matched domain pair (source and target concepts) is used. This same approach is used to compute the range similarity measure. When the ranges of two data type properties are being compared, the semantic similarity measure proposed by Wu and Palmer [23] is used:

$$Sim_{WuP}(\mathcal{R}_s, \mathcal{R}_t) = \frac{2 \times \mathcal{N}}{\mathcal{N}_s + \mathcal{N}_t + 2 \times \mathcal{N}} \quad (4)$$

where \mathcal{N}_s and \mathcal{N}_t are the number of IS-A links between the ranges from the source and target properties to their closest common ancestor XML data type P ; \mathcal{N} is the number of IS-A links from P to the root of the XML data type hierarchy. Regardless of the ontology language, Sim_{WuP} can be used to find the distances between ranges as long as the allowable values for the ranges of properties form a taxonomy structure; otherwise, the measure will be either 1 or 0. The final internal similarity measure is calculated by combining type, domain, and range measures using equation (1).

2.4 Individual Similarity (Sim_N)

ASMOV does not rely on individuals to find the similarity between concepts and properties; however, if individuals are present, the similarity between them is exploited in order to refine the mapping. Two individuals are said to be similar if their internal structures are the same and the corresponding literal values match. If both of these requirements are met for one or more individuals of the source and target entities, then the similarity measure between these entities is set to 1. Moreover, no additional calculations are performed between those entities. The individual similarity between properties is more complex. As the concepts' individuals are analyzed, their internal structures will highlight the similarity between properties. In some cases, individuals may have missing properties; these missing properties are analyzed in order to update a list of possible matches between properties. If the possible matches between the source properties and target properties are a one-to-one mapping, then the similarity between those properties are also set to 1; on the other hand, the similarity measure is the probability calculation. Thus, the individual similarity between properties is dictated by the number of occurrences of the target property within the possible matches and the total number of items the list of possible matches.

3. ALIGNMENT ALGORITHM

ASMOV introduces an automated, rule-based approach for the alignment of two ontologies that exploits the universality of knowledge about each ontology and semantically validates the potential alignment at the end of each iteration.

The ASMOV alignment procedure is illustrated by the pseudo-code shown in Figure 1, which highlights its five stages. In the initial stage, users are provided with the option of supplying initial mappings to the system. The system will regard these mappings as facts. These facts, as long as they are validated by our mapping validation process (section 3.3), are not challenged during the iterations that follow. When the partial alignment provided by the user is not accurate, he/she is warned with appropriate feedback from the system and has the opportunity to rectify the issues or specify that the system ignore the inconsistencies.

```

load ontology  $\mathcal{O}_1, \mathcal{O}_2$ 
-- preprocessing
adjust weight depending on missing attributes
for each entity  $\mathcal{E}_1 \in \mathcal{O}_1, \mathcal{E}_2 \in \mathcal{O}_2$ 
  sort  $\mathcal{E}_1, \mathcal{E}_2$  parent-first
  -- text processing
  for each lexical attribute  $\mathcal{L}_{a1} \in \mathcal{E}_1, \mathcal{L}_{a2} \in \mathcal{E}_2$ 
    tokenize text into words
    for each word
      retrieve and store synonyms
-- compute similarities between  $\mathcal{O}_1, \mathcal{O}_2$ 
create two-dimensional similarity matrix,  $\mathcal{M}$ ,
  between entities in  $\mathcal{O}_1, \mathcal{O}_2$ 
do
  while total change in  $\mathcal{M} > \lambda$ 
    for each entity  $\mathcal{E}_1 \in \mathcal{O}_1, \mathcal{E}_2 \in \mathcal{O}_2$ 
      update  $\mathcal{M}[\mathcal{E}_1, \mathcal{E}_2]$  with similarity between  $\mathcal{E}_1, \mathcal{E}_2$ 
    -- prune matrix of dissimilar entities
    for each entity  $\mathcal{E}_1 \in \mathcal{O}_1, \mathcal{E}_2 \in \mathcal{O}_1$ 
      resolve crisscross or many-to-one issues.
  while mapping validation is unsuccessful
-- ontology creation
create alignment  $\mathcal{A}$  between  $\mathcal{O}_1$  and  $\mathcal{O}_2$ 

```

Figure 1. ASMOV Pseudo Code

In the second stage, the source and target ontologies are loaded into memory and are then pre-processed. This pre-processing phase has three dimensions:

Weight Adjustment: The four key features in an ontology and the ones that take part in the partial similarity calculations have different impacts on the probability that two entities should be mapped between ontologies; however, not all ontologies in practice contain every feature. In these cases, the corresponding weights, or importance, are adjusted based on the presence or absence of the features within the ontology. The initial weights and their relative adjustments were determined by optimizing ASMOV against the OAEI 2006 benchmark tests [19].

Textual Pre-processing: As ASMOV relies on a lexical reference system (a thesaurus) for the retrieval of semantic relations between words, the identifiers of the entities within the ontologies must be tokenized. This task is necessary as identifiers may be a combination of multiple words. One such example is PhdThesis; the tokenization process will produce the two tokens: Phd and Thesis. Ontologies such as OWL contain annotation properties (labels and comments) that provide additional descriptive information about the entities to which they belong. The values of

these properties are pre-processed as well and are used in the lexical similarity calculations. The semantic relationships utilized by ASMOV include hypernym, antonyms and synonyms.

Ordering: When calculating the similarity between two entities, the similarity measure between their parents is taken into account. By re-ordering the entities in a way such that the parent entities are processed first, we reduce the number of iterations, while insuring that the similarity measures between parent entities are available when comparing their sub-entities. This improves both the performance and the accuracy of the algorithm.

In the third stage, ASMOV iteratively computes the similarities among the concepts and properties (Section 2), while storing the results in a 2-dimensional similarity matrix, \mathcal{M} . The iterative process will only stop once the overall change in the matrices for two subsequent iterations is below a given threshold, λ . A threshold of zero means that the iterative process will stop only if two consecutive iterations produce the same matrices. Setting a higher threshold may reduce the number of iterations necessary to complete the alignment process, but it may also decrease the accuracy of the alignment. The iterative process allows ASMOV to identify complex interdependencies and relationships between entities. For example, a concept has properties, and a property is defined by its domain which can either be a concept or a collection of concepts.

The fourth stage is performed at the end of each iteration by applying rules to its temporary alignment in order to discover and analyze semantic inconsistencies (section 3.1). Such inconsistencies include both many-to-one and crisscross mappings which are pruned if they cannot be justified.

The final stage is performed after a check for convergence (section 3.2) where the resulting alignment is validated against the semantic meaning of the ontologies (section 3.3). If any mapping is found to be incorrect, it is placed in a list of erroneous mappings and the iteration process starts again, otherwise the process completes and returns the set of validated alignment.

3.1 Pruning Process

The pruning process occurs at the end of each iteration in order to remove any incorrect or invalid mappings and to resolve any semantic issues.

Removal of incorrect and invalid mappings: An invalid mapping is a source-target pair of concepts that if it remains in the alignment it would cause semantic conflicts. Invalid mappings are discovered at the end of each iteration and are maintained in a list. As changes in the similarity measures occur in subsequent iterations, the list of invalid mappings is updated to remove inconsistencies.

Resolution of semantic issues: the resolution of semantic issues, such as crisscross mapping and multiple-entity mapping, is driven by a set of rules. A crisscross mapping occurs whenever a source entity (\mathcal{SE}_p) and its child (\mathcal{SE}_c) are respectively mapped to a target entity (\mathcal{TE}_c) and its parent (\mathcal{TE}_p). In this case the confidence values of the mappings are analyzed. If the \mathcal{SE}_p and \mathcal{TE}_c pair has a higher confidence value, the \mathcal{TE}_p and \mathcal{SE}_c pair is deemed invalid and is added to the invalid mapping list. In subsequent iteration, \mathcal{TE}_p and \mathcal{SE}_c will never be mapped as long as \mathcal{SE}_p and \mathcal{TE}_c are mapped.

Multiple-entity mapping occurs when multiple source entities are mapped to the same target. Given two entities \mathcal{SE}_1 and \mathcal{SE}_2 from the source ontology being mapped to the same target entity \mathcal{TE} . Before considering the confidence values in order to decide which mapping is correct, an equivalency check is made between \mathcal{SE}_1 and \mathcal{SE}_2 . There are two types of equivalencies that can be discovered: explicit and implicit. Explicit equivalencies are defined through the use of equivalent constructs such as *equivalentClass* and *equivalentProperty* in OWL. An implicit equivalency arises when \mathcal{SE}_1 and \mathcal{SE}_2 have a parent-child relationship and the child concept does not add any more knowledge than its parent to an ontology. When the two entities \mathcal{SE}_1 and \mathcal{SE}_2 are not equivalent, the mapping with the lowest confidence value is added to the invalid mapping list.

3.2 Convergence

Since the iterative process of the alignment algorithm may never converge on its own, a checksum is calculated for the alignment produced at the end of each iteration. These alignments are built by pulling the best matches of the source ontology's concepts and properties from the similarity matrices. The alignment is an ordered sequence of triples, each consisting of source entity, target entity and confidence value. A hash function is used to compute the checksum, which is maintained throughout the mapping process. Whenever a checksum is the same as one calculated in a prior iteration, the mapping process stops. Although this process ensures convergence, it may be costly, as the possible number of unique checksums depends on the number of concepts and properties contained in the two ontologies as well as the precision of the confidence value. The pruning process and the mapping validation step of the algorithm help ensure convergence. This allows our algorithm to avoid cycles of incorrect iterations that may result in a poor precision and/or recall of the alignment. It also reduces the number of possible checksums after each iteration.

3.3 Mapping Validation

After a series of iterations converge, the mapping validation begins. This process performs a structure analysis on a graph built from the alignment and information from the ontologies. Two different constructs constitute this graph: nodes and edges. The nodes contain pairs of entities, whereas the edges contain pairs of properties. The validation process is done in three phases: concept validation, property validation, and concept-property validation. In the first two phases, the edges considered are created using the predefined properties of the ontology. Currently, three types of edges are supported: IS-A, SAME-AS, and DISJOINT-FROM. The IS-A edge illustrates the parent-child relationships between entities; the SAME-AS edge specifies the equivalence relationships between nodes; and the DISJOINT-FROM edge points out the nodes that should not have any overlap. These edges are added to the graph by examining both the source and target ontologies. The validation of the graph can then be reduced to an investigation of edge violations; a node may not be valid if one or more of the edges are violated. Since an edge links two nodes, the complexity here is to select the proper node that needs to be invalidated. The edges of type IS-A will take precedence over the others in the validation process. To that effect, as long as a node has a valid IS-A edge, it will not be invalidated. The validation process does not take any action in the case where two nodes with valid IS-A edges are linked by an edge which presents a violation, which may occur when the ontologies themselves are

inconsistent. Nodes and edges may not be fully qualified; in other words, they may be missing one of their entities. For example, a source concept that is not mapped to any target concept will form non-fully qualified nodes. If an edge violation exists, only the linked nodes are investigated. An edge may not be fully qualified as well. For instance, the source entity within a node may have a parent, which may not be true for the target entity, thus creating an incomplete IS-A edge. When an incomplete edge is present and a violation exists, the node with the lowest confidence value is invalidated.

In the third phase, the concept validation graph is modified. All edges are dropped from the remaining valid nodes and are replaced by edges created from the valid nodes of the property validation graph. The new graph is then validated, but this time the nodes are favored; thus, only the edges get invalidated. All invalid mappings that have been identified are added to the invalid mapping list. If at least one violation was identified, the iteration process resumes and the invalid source-target pairs are ignored.

4. EXPERIMENT AND RESULTS

The 2006 benchmark series of tests created by the Ontology Alignment Evaluation Initiative (OAEI) [19] have been used to identify the strengths and weaknesses of our alignment algorithm. With these series of tests we are also able to compare our system to others mentioned in the introduction section. Although the tests are confined to the domain of bibliographic references (BibTeX), ASMOV works with ontologies in other domains as well. The benchmark tests start from a reference ontology to a multitude of alterations (some less likely to occur in a real world situation). As ontologies may be modeled in the different manner by different developers, the variations between the tests highlight how well the algorithm would perform in the real world.

4.1 ASMOV Implementation

We have implemented ASMOV as a Java application. ASMOV relies on Jena [12] to parse and load the ontologies from RDFS and OWL files. For the lexical similarity, ASMOV interfaces with the Java WordNet library (JWordNet) [13] in order to query for the hypernyms, antonyms and synonyms. Since the current implementation of JWordNet is only compatible with WordNet 2.0, it was updated in order to be compatible with the latest version of WordNet for the windows environment (version 2.1). The library was also restructured in order to be generic enough to interface with other thesauri.

4.2 Experimental Setup

The experiment was carried out on a PC running Windows XP Professional with a dual-core Intel Pentium processor (2.8 GHz) and 3 gigabytes of memory. A few adaptations had to be made for evaluation purposes. First, a mechanism to produce the alignment in the format dictated by OAEI had to be added to the implementation of ASMOV. Second, our system favors individual similarity over textual similarity, which is not the case according to the gold standard alignments. Thus, we have reversed that assumption for the evaluation.

4.3 Analysis of Results

The accuracy (in terms of precision, recall, and F1-measure) of ASMOV is compared to the systems that produced the best results at the OAEI 2006 campaign (COMA, Falcon and RiMOM). Also, due to the similarity in approaches between OLA and ASMOV, we compare our results to accentuate the differences between the

systems. The precision and recall for all systems are extracted from an evaluation tool supplied by OAEI. In order to compare the systems based on a single measure, we have added an F1-measure given by the equation:

$$F = \frac{2 \times R \times P}{R + P} \quad (5)$$

Where R and P are respectively the recall and the precision. The results of running ASMOV, Coma, Falcon, and RiMOM against the OAEI 2006 benchmark tests for are presented in Table 1.

Table 1. Results of OAEI 2006 Tests

	ASMOV	Coma	Falcon	RiMOM
P	93%	96%	92%	96%
R	90%	83%	86%	88%
FI	91.5%	89%	88.9%	91.8%

The OAEI 2006 benchmark tests include a series of tests (248-266) that are unlikely to exist in real-world ontologies, where it is difficult to recognize the correct alignment [11] Specifically, these ontologies replace meaningful words with random characters, textual features may be removed completely, and the semantic links between concepts are removed. As a result, all systems performed relatively poorly in these tests. Table 2 compares ASMOV, Coma, Falcon, and RiMOM against only the practical tests of the OAEI benchmarks.

Table 2. Results of OAEI 2006 Practical Tests

	ASMOV	Coma	Falcon	RiMOM
P	97%	97%	97%	98%
R	98%	94%	85%	97%
FI	97.5%	95.5%	90.6%	97.5%

Finally, since the only results published by OLA [5] are from the EON 2004 benchmark tests, we ran ASMOV against the same tests in order to have an accurate comparison. For these tests, the overall precision, recall, and F1-measure were calculated as an average of the values obtained from individual tests. OLA published two sets of results, one where the weights are adjusted on a test by test basis (OLA₁) and the other where all tests use the same weight distribution (OLA₂). The results of running ASMOV and OLA against the EON 2004 benchmark tests are presented in Table 3.

Table 3. Results of EON 2004 Tests

	ASMOV	OLA ₁	OLA ₂
P	92%	84%	82%
R	95%	53%	51%
FI	93.7%	63.1%	62%

4.4 Discussion

The results from Table 1 show that ASMOV performs as well as the top three systems in all of the OAEI 2006 benchmark tests. Comparing the systems against only those tests that are likely to occur in the real-world show that ASMOV performs better than both Coma and Falcon and achieves an identical F1-measure as RiMOM, as shown in Table 2. And, while OLA and ASMOV share many of the same ideas for the alignment of ontologies, the results from Table 3 show that ASMOV outperforms OLA, even when the weights in OLA are manually tuned to the specific test. In the tests for Table 2, there is a difference between the results published by Falcon and those that were produced by the OAEI tool as the OAEI tool did not return results for three of the tests containing UTF8 characters.

5. CONCLUSION AND FUTURE WORK

In this paper we introduce ASMOV, a novel approach to the ontology alignment realm. Experimentation has shown that ASMOV performs as well as the top three systems in the OAEI 2006 benchmark tests and significantly better than OLA in the EON 2004 benchmark tests. However, we believe that ASMOV can be improved. First, although the weights chosen for the similarity measures should be the same regardless of the domain of the ontologies, we need to perform more tests in order to validate this and fine tune the weights. Second, convergence needs to be revisited as the current solution may cause the iterative process to end prematurely and thus produce a less than optimum alignment. Third, the performance of the current lexical reference system (WordNet) is slower than those used by other systems. Others such as RiMOM have used external packages to handle these types of calculations more quickly; we need to either optimize our implementation or interface with a different external package. Finally, ASMOV need to be extended to be able to present the user with a graphical interface, facilitating system-user interaction.

6. ACKNOWLEDGEMENTS

This work is funded by the National Institutes of Health (NIH) under grant R43RR018667. The authors also wish to acknowledge the contribution of Mr. Thomas Taylor, Mr. Mike Ryan, and Mr. Patrick Shironoshita of INFOTECH Soft, Inc.

7. REFERENCES

- [1] Bach, T., Dieng-Kuntz, R. Measuring Similarity of Elements in OWL-DL ontologies, *In American Association for Artificial Intelligence*, 2002
<http://dit.unitn.it/~pavel/cando/Pictures/Posters/WS105BachTL.pdf>
- [2] Dieng, R. and Hug, S. Comparison of "personal ontologies" represented through conceptual graphs. *In Proc. 13th ECAI*, Brighton (UK), 1998, 341–345.
- [3] Doan, A., Madhavan, J., Domingos, P., and Halevy, A. Learning to map between ontologies on the semantic web. *In The Eleventh International WWW Conference*, Hawaii, US, 2002.
- [4] Euzenat J. and Valtchev P. Similarity-based ontology alignment in OWL-lite. *In Proc. 15th ECAI*, Valencia (ES), 2004, 333-337.
- [5] Euzénat J., Loup, D., Touzani, M., Valtchev P., "Ontology Alignment with OLA", to appear in *Proceedings of the 3rd EON Workshop*, 3rd Intl. Semantic Web Conference, Hiroshima (JP), November 2004.
- [6] Euzenat, J. Brief overview of T-tree: the Tropes taxonomy building tool. *In Proc. 4th ASIS SIG/CR workshop on classification research*, Columbus (OH US), 1994, 69–87.
<ftp://ftp.inrialpes.fr/pub/sherpa/publications/euzenat93c.pdf>.
- [7] Euzenat, J. *State of the art on ontology alignment*.
<http://www.inrialpes.fr/exmo/cooperation/kweb/heterogeneity/deli/>. August, 2004.
- [8] Giunchiglia, F, Shvaiko P., and Yatskevich, M. S-Match: An algorithm and implementation of semantic matching. *In Proceedings of the European Semantic Web Symposium, LNCS 3053*, pp.61-75, 2004.
- [9] Giunchiglia, F. and Shvaiko, P. Semantic matching. *In The Knowledge Engineering Review journal*, 18(3): 2004, 265-280
- [10] Hu, W. and Qu, Y.: Block Matching for Ontologies. *Proceedings of the 5th International Semantic Web Conference (ISWC 2006)*.
- [11] Hu, W., Cheng, G., Zheng D., Zhong X., and Qu, Y: The Results of Falcon-AO in the OAEI 2006 Campaign. *International Workshop on Ontology Matching (OM 2006)*
- [12] Jena from HP Labs <http://www.hpl.hp.com/semweb/>
- [13] JWordNet
<http://www.seas.gwu.edu/~simhaweb/software/jwordnet/>
- [14] Lin, D.. An Information-Theoretic Definition of Similarity. *In 15th International Conference on Machine Learning*, Morgan Kaufmann, 1998, 296--304.
- [15] Massmann, S.; Engmann, D.; Rahm, E.: COMA++: Results for the Ontology Alignment Contest OAEI 2006. *International Workshop on Ontology Matching, collocated with the 5th ISWC-2006*; Athens, Georgia, USA, 2006
- [16] Noy, N. F. and 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-2001)*, Seattle, WA, 2001.
- [17] Noy, N. F. and Musen, M. A.. PROMPT: Algorithm and tool for automated ontology merging and alignment. *In Seventeenth National Conference on Artificial Intelligence AAAI (2000)*, 450-455.
- [18] Rahm, E. and Bernstein, P. A. A survey of approaches to automatic schema matching. *The VLDB Journal*, 2001, 10:334-350.
- [19] Shvaiko, P., Euzenat, J., Stuckenschmidt, H., Yatskevich, M., van Hage, W. R., Mochol, M., Svatek, V. *Ontology Alignment Evaluation Initiative Test library, 2006*
<http://oei.ontologymatching.org/2006/benchmarks/>
- [20] Stumme, G. and M'adche, A. FCA-Merge: Bottom-up merging of ontologies. *In 7th Intl. Conf. on Artificial Intelligence (IJCAI '01)*, Seattle, WA, 2001, 225–230.
- [21] Tang, J., Li, J., Liang, B., Huang, X., Li, Y., and Wang, K. Using Bayesian Decision for Ontology Mapping. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, Vol 4, issue 4 (December 2006), 243-262
- [22] WordNet <http://wordnet.princeton.edu/>
- [23] Wu, Z. and Palmer, M. Web semantics and lexical selection. *In Proceedings of the 32nd Annual Meeting of the Associations for Computational Linguistics*, 1994, 133-138