Combination of Similarity Measures in Ontology Matching using the OWA Operator

Qiu Ji, Peter Haase, Guilin Qi
Institute AIFB
University of Karlsruhe
D-76128 Karlsruhe, Germany
{qiji,pha,gqi}@aifb.uni-karlsruhe.de

Abstract

In this paper, we provide a novel solution for ontology matching by using the ordered weighted average (OWA) operator to aggregate multiple values obtained from different similarity measures. We have implemented the solution in the ontology matching system FOAM. Using the base matchers in FOAM, we analyze the way to choose different OWA operators and compare our system with others.

1 Introduction

Ontology matching aims at identifying correspondences between elements in multiple ontologies. Ontology matching has many application areas, such as data integration, data merging, and semantic search across heterogeneous data sources. So far, quite a number of ontology matching systems have been proposed. Good surveys of different approaches to the matching problem are provided in [6, 7]. Most approaches rely on similarity-based techniques that try to find correspondences based on various similarity measures, each computed by individual base matchers.

It has been accepted that combining the results of multiple base matchers is a promising technique to obtain more accurate matching results than just using one matcher at a time. Usually, a simple weighted average is used as the aggregation function where the weights can be obtained manually or by machine learning techniques. Obviously, it is difficult for a person to manually assign the weights by experience. Conversely, for methods based on machine learning method, rich data sets are needed to train the algorithms to obtain useful weights.

To alleviate this problem, we investigate the use of the *Ordered Weighted Average* (OWA) for the aggregation of similarity values obtained by individual similarity measures. A weight used by the OWA operator is associated not with a specific similarity measure, but instead with a specific *ordered position*. We have implemented our solution in the ontology matching system FOAM[1]. There are two main reasons for integrating the OWA operator into FOAM:

1. FOAM provides many base matchers according to various features of OWL ontologies such as super-concepts and sub-concepts.
2. The OWA operator is a powerful operator to aggregate multiple values, and there are many kinds of approaches to obtain OWA weights. Particularly, the linguistic OWA operator provides semantic explanations which can be understood by users easily.

This paper is organized as follows: In Section 2, we discuss related work on aggregation of similarity measures in ontology matching. In Section 3 we describe the background of

[1] http://ontoware.org/projects/map

L. Magdalena, M. Ojeda-Aciego, J.L. Verdegay (eds): Proceedings of IPMU'08, pp. 243–250
Torremolinos (Málaga), June 22–27, 2008

the FOAM system and the OWA operator and then discuss the integration of OWA operator into FOAM and the resulting problems. We evaluate the OWA combination method based on FOAM in Section 4. In Section 5 we conclude and give an outlook to future work.

2 Related Work

Ontology matching systems have been developed by many researchers [6, 7]. They consider various kinds of information provided in ontologies. To aggregate results of multiple base matchers, many combination methods have been proposed. In the following we discuss the ones are most related to our work.

COMA [1] exploits *Max*, *Min*, *Average* and *Weighted* strategies for combination. The *Weighted* strategy needs a relative weight for each matcher to show its relative importance. The study in paper [3] uses linear (weighted) combination to aggregate the similarities obtained by all the features that make the definition of an entity in an OWL-Lite ontology.

CMC [10] combines multiple schema matching strategies based on credibility prediction. It needs to predict the accuracy of each matcher on the current matching task first by a manual rule or a machine learning method. Accordingly, different credit for the matcher is assigned. From each base matcher, two matrices including the similarity matrix and the credibility matrix are provided. The credibilities are used as weights to aggregate multiple similarity values obtained by the base matchers into a single one.

In the original version of FOAM, which implements the algorithms described in [2], both manual and automatic approaches to learn how to combine the similarity values are provided.

To sum up, the *weighted average* is the most popular aggregation operator to combine multiple values. The weight here is assigned to different base matchers and can be obtained manually or by machine learning techniques. When it is not necessary or difficult to get weights for base matchers, only *Max*, *Min* and *Average* can be used. However, since each base matcher performs differently under different conditions, these operators may be not enough to show various performance for complex situations.

In our previous work [4], linguistic OWA operators are introduced to aggregate multiple values for ontology matching. But there is no details about

1. How the performance of OWA operator behaves with more base matchers and complex ontologies.
2. How to choose different combination operators for different purposes.
3. How the performance of the system compares with other ontology matching systems.

In this paper, we provide a new solution for ontology matching by integrating OWA operator into the FOAM system. Based on this solution, the questions above can be answered accordingly.

3 The OWA Operator for Ontology Matching

3.1 Ontology Matching in FOAM

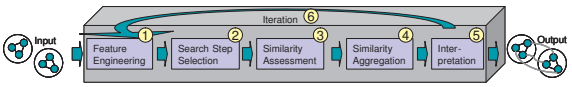

Figure 1: Ontology matching process in FOAM

The FOAM system is based on a generic process for ontology matching, as described in [2]. Other ontology matching approaches can be described in terms of this process as well. Here, we only describe briefly the process to the extent that is necessary to understand how the role of similarity aggregation works within this process. Figure 1 illustrates the six main steps of the generic process.

Input: Input for the process are two or more ontologies, which need to be matched

with one another. Additionally, it is often possible to enter pre-known (manual) matches. They can help to improve the search for matches.

1. Feature engineering: The role of feature engineering is to select relevant features of the ontology to describe a specific ontology entity, based on which the similarity with other entities will later be assessed. For instance, the matching process may only rely on a subset of OWL primitives. For each feature, a specific matcher based on a similarity measure will be assigned.

2. Search Step Selection: The derivation of ontology matches takes place in a search space of candidate matches. This step may choose to compute the similarity of certain candidate element pairs and to ignore others in order to prune the search space [2].

3. Similarity Computation: For a given description of two entities from the candidate matches this step computes the similarity of the entities using the selected features and corresponding similarity measures.

4. Similarity Aggregation: In general, there may be several similarity values for a candidate pair of entities, e.g., one for the similarity of their labels and one for the similarity of their relationship to other entities. These different similarity values for one candidate pair have to be aggregated into a single aggregated similarity value. Often – as in the original FOAM system – a weighted average is used for the aggregation of similarity values.

5. Interpretation: The interpretation finally uses individual or aggregated similarity values to derive matches between entities. A common approach is to use thresholds [1]: If the similarity of two elements exceeds the threshold, the elements are considered as a match.

6. Iteration: The similarity of one entity pair influences the similarity of neighboring entity pairs, for example, if the instances are equal this affects the similarity of the concepts and vice versa. Therefore, the matching process is repeated until no new alignments are proposed or a fixed number (for our test, the maximal iteration is set 3) of iteration is reached.

Output: The output is a representation of matches and possibly with additional confidence values based on the similarity of the entities.

3.2 The Ordered Weighted Average Operator

The ordered weighted averaging (OWA) operator is introduced in [11] to aggregate information. It has been used in a wide range of application areas, such as neural networks and fuzzy logic controllers.

Assume we are given a set of arguments $V_1 = (a_1, a_2, ..., a_n)$, $a_i \in [0,1]$, $1 \leq i \leq n$, and the weights for OWA operator $\mathrm{W} = (\mathrm{w}_1, ..., \mathrm{w}_n)$. After reordering the elements in V_1 in descending order, we mark it as $V_2 = (b_1, b_2, ..., b_n)$, where b_j is the j_{th} highest value in V_1. An OWA operator is a mapping function F from I^n to I, $I = [0,1]$:

$$F(a_1, a_2, ..., a_n) = \sum_{i=1}^{n} \mathrm{w}_i b_i$$
$$= \mathrm{w}_1 b_1 + \mathrm{w}_2 b_2 + ... + \mathrm{w}_n b_n,$$

where $\mathrm{w}_i \in [0,1]$ and $\sum_{i=1}^{n} \mathrm{w}_i = 1$.

Note that a weight w_i is not associated with a particular argument a_i, but with a particular ordered position i of the arguments. That is w_i is the weight associated with the ith largest argument whichever component it is [11].

Obviously, determining the OWA weights w_i, $1 \leq i \leq n$ is a critical task. So far, quite a few approaches have been proposed. We adopt the linguistic quantifiers developed by Yager [11], since these quantifiers have semantics which can be accepted easily for users. They are defined as:

$$w_i = Q(i/n) - Q((i-1)/n), i = 1, 2, ..., n \quad (1)$$

where Q is a nondecreasing proportional fuzzy linguistic quantifier and is defined as the fol-

lowing:

$$Q(r) = \begin{cases} 0, & \text{if } r < a; \\ (r-a)/(b-a), & \text{if } a \le r \le b; \\ 1, & \text{if } r > b, \end{cases} \quad (2)$$

where $0 \le a, b, r \le 1$, a and b are the predefined thresholds. Obviously, the operators such as *Maximal*, *Minimal* and *Average* are three special cases of OWA operator.

For some special linguistic operators like *at least half* which are used in the paper, we introduce their semantic interpretations within the application area of ontology matching to facilitate users to choose different operators for different tasks or purposes.

Assume there are n base matchers m_1, m_2, ..., m_n. Each base matcher can be regarded as a criteria, so the aggregation process is to form an overall decision by considering multiple criteria. For an entity pair (x, y), where x belongs to source ontology and y belongs to target ontology, $m_i(x, y)$ indicates the degree to which the entity pair (x, y) satisfies the criteria or base matcher m_i. Actually, $m_i(x, y)$ is the similarity value between x and y obtained by base matcher m_i, where i= 1, 2, ..., n.

1. Maximal: $Max(x, y) = Max\{m_1(x, y), m_2(x, y),...,m_n(x, y)\}$, where *Max* means that (x, y) satisfies at least one of the matchers, i.e., satisfies m_1 or ... or m_n.

2. Minimal: $Min(x, y) = Min\{m_1(x, y), m_2(x, y), ..., m_n(x, y)\}$. *Min* means that (x, y) satisfies all the matchers, that is to say, we are essentially requiring to satisfy m_1 and ... and m_n.

3. Average: *Average* means identity, which regards all similarity values equally.

4. At least half: This operator satisfies at least half matchers. Actually, it only considers the first half of similarity values after re-ordering them in descending order.

5. Most: *Most* means most of the matchers is satisfied. Usually, this operator ignores some higher and lower similarity values, that is to give small weights on them, while paying more attention to the values in the middle of the input arguments after re-ordering.

6. As many as possible: It satisfies as many as possible matchers and is opposite to *at least half*. The second half of values after reordering is considered. So after an aggregation operation, the result obtained by *at least half* is always higher than that by *as many as possible*.

3.3 Integration of OWA into FOAM

In FOAM, originally a *weighted average* is used for the similarity aggregation step, which gives more importance to the labels of the entities to be compared. If there is no label available, the performance will become worse.

In our work, we use the OWA operator to combine similarity values obtained from multiple similarity methods. Obviously, the main advantage is that no weights are fixed to these similarity methods, but to the positions of these similarity values in a descending order. In this way, each base matcher is treated equally. The second advantage is that, although the OWA weights can be obtained manually or by machine learning techniques, there are quite a few straightforward methods without data sets for training and too much preliminary knowledge. What the users need to do is to take several ontology pairs to be compared as samples and observe the results obtained by the base matchers for each entity pair, or they can simply choose different linguistic OWA operators by their semantic interpretations.

To choose OWA operators by observation, we assume there are n single base matchers for a category, which can be concept category, data property category, object property category or instance category. If most single matchers could return m ($0 \le m \le n$) similarity values sim_i above zero, where $0 \le i \le m$, and $0 < sim_i \le 1$, then it is better to choose an OWA operator which can give some importance to most of the m highest values or all of them, while assigning lower or zero to other $n - m$ values. Basing on the base matchers

we use in this paper, no more than half of the base matchers return some similarity values above zero in most cases. Based on our experience, higher values are more reliable, but one should not rely on just one highest value. So it would be better to use *at least half* which only considers the half higher values, but not to use *maximal* considering one extreme value for each aggregation.

4 Evaluation and Discussion

4.1 Data Sets

We use the benchmarks which are provided by the OAEI campaign in 2006[2]. The benchmarks test case includes 51 ontologies in OWL. Ontology 101 is regarded as the reference ontology, i.e., each ontology in the benchmarks, including ontology 101, will be matched against it. The benchmarks are divided into three groups marked as 1xx (ontology 101-104), 2xx (ontology 201-266) and 3xx (ontology 301-304). More details can be found on the website of OAEI 2006.

The goal of this benchmark series is to identify the areas in which each matching algorithm is strong and weak. For testing, the results are computed automatically without the participation of users. We also obey the rule to obtain mappings to compare with other systems in OAEI 2006.

4.2 Evaluation Criteria

In order to compare the performance of different matching algorithms, several evaluation criteria are used to give different views of the results. Except the standard measures such as *precision*, *recall* and *f-measure*, the harmonic mean measure is also used to compare our results with those provided by other ontology matching system in OAEI 2006.

For the measures below, i indicates the i_{th} test. $|R_i|$ refers to the number of correct mappings according to the gold standard (which is manually created). $|P_i|$ is the total number of mappings found automatically by the matching system, and $|I_i|$ is the number of *correct mappings* found by the matching system for test i.

1. Precision (p): $p_i = |I_i|/|P_i|$. It reflects the ratio of the correct mappings among all mappings discovered by the matcher.

2. Recall (r): $r_i = |I_i|/|R_i|$ specifies the ratio of correct mappings found by the matcher in comparison with total number of mappings in the golden standard.

3. F-Measure (f): $f_i = 2 * p_i * r_i/(p_i + r_i)$, which estimates the reliability of the match predictions [1].

4. Harmonic mean (H): Harmonic mean[2] is an aggregation of standard measures such as p, r. Specifically, for harmonic mean of *precision*, $H(p)$=$(\sum_{i=1}^{n} |I_i|)$ / $(\sum_{i=1}^{n} |P_i|)$. As for harmonic mean of *recall*, $H(r) = (\sum_{i=1}^{n} |I_i|)$ / $(\sum_{i=1}^{n} |R_i|)$. Here n is the number of considered tests. We then obtain for the harmonic mean of the F-Measure: $H(f) = 2 * H(p) * H(r)/(H(p) + H(r))$.

4.3 Results and Discussion

We use 23 base matchers such as entity label, super concepts and sub-concepts. These matchers are provided by FOAM and can be found in the class "ManualRuleSimple" in FOAM API.

4.3.1 The performance of OWA operators

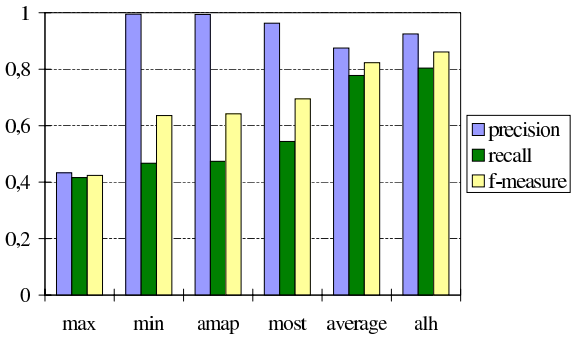

Figure 2: The performance of different OWA combination methods

[2]http://oaei.ontologymatching.org/2006/

In the first part of the evaluation, we compare the performance of different OWA operators for the aggregation of similarity values provided by the base matchers in FOAM.

Figure 2 shows the harmonic means (over the entire OAEI 2006 benchmarks data set) of the precision, recall and f-measure for the different operators introduced in Section 3.3.

For these OWA operators, they assign the importance to different positions of the values to be aggregated in the descending order. For example, *max* and *min* consider the extreme values, maximal value or minimal value respectively. The first observation is that the *min* and *max* operators show a poor performance, as they assign all the weights to only one matcher. However, the *min* operator exhibits a very high precision, as it will only return a match if even the matcher with the smallest similarity value indicates a match. Obviously, this selectivity results in a low recall. *As many as possible* shows a slightly better (but similar) performance as *min*, as it also assigns most weights to the matchers with low similarity values. We observe an increasing performance in terms of f-measure for the operators from *most*, the *average*, to *at least half*. The best results in terms of f-measure are obtained for the *at least half* operator that assigns the weights to that half of similarity values which are the highest ones.

However, it is worth noting that for different matching tasks, different operators may be appropriate. For example, if a high precision is required, an operator that assigns more weight to the lower similarities may be adequate, e.g. *as many as possible*. In any case, this selection can be performed easily based on the intuitive meaning of the lexical OWA operators, without any knowledge about the specific base matchers.

4.3.2 Comparison between OWA operator and weighted average

In the second part of the evaluation, we compare the performance of OWA operator with that of the regular *weighted average*, which was previously implemented in the FOAM system. Again, the comparison is done based on exactly the same base matchers.

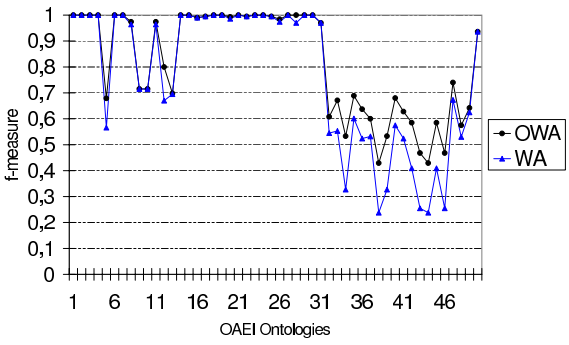

Figure 3: The performance of *at least half* and *weighted average*

In Figure 3, we show the f-measure of all the ontology pairs. Where the number from 0 to 50 in X-axis indicates the ontology pair between ontology from 101 to 304 and the reference one 101. For each pair, one element is in the reference ontology 101, and another one is in the test ontology from 101 to 304. Two curves indicate the f-measure which are calculated by the *at least half* OWA operator and *weighted average* respectively.

We observe almost no difference in the f-measure between *at least half* and the *weighted average* for the "easy" cases, i.e. the ontologies with large overlap. In these cases, the f-measure is close to 1.

In the harder cases, e.g. ontologies from 32 to 46 (corresponding to OAEI ontologies from 248 to 266), many places have been changed based on the reference ontology 101 comparing with others. For example, all entity names in these ontologies have been replaced by random strings, and there is no comment available for the entities. Parts of the class hierarchy have been suppressed, expanded or flattened. So the structure between these ontologies and the reference one are quite different.

In such cases, some base matchers will become useless if the corresponding features are not available. The *weighted average* is not flexible enough to deal with such cases since it gives weights to each base matcher independent of the performance of the matcher. On the other

hand, *at least half* does not rely on some particular base matchers. Instead, it will assign the weights to that half of the matchers that perform best. On average, we observe an increase in for the f-measure from 0.82 for the weighted average to 0.86 for the *at least half* OWA operator.

The benefits of using OWA combination operators are actually twofold: We do not need to assign weights to the individual base matchers thus do not require any background knowledge about the base matchers. At the same time we observe an improved performance.

4.3.3 Comparison with other ontology matching systems

Although the FOAM system did not participate in the official evaluation contest of OAEI in 2006, we give our results to compare with other systems[2], based on the same benchmarks with other systems and using the same evaluation measures. Since many systems attended the contest, we only compare against the top five systems.

System	H(p)	H(r)	H(f)
automs	0.94	0.67	0.782
COMA++	0.96	0.83	0.890
falcon	0.92	0.86	0.889
prior	0.95	0.63	0.758
RiROM	0.96	0.88	0.918
FOAM-WA	0.87	0.78	0.823
FOAM-OWA	0.93	0.80	0.860

Table 1: The comparison between FOAM (WA and OWA) and other matching systems based on the benchmarks in OAEI 2006.

In Table 1, we show the harmonic means of precision and recall and f-measure to give an overview of the results. From the results shown in this table, we see that three matching systems, coma, falcon and RiROM are relatively close, and that the FOAM system – with either weighted average or ordered weighted average – provides competitive performance. And it can be seen that the test using OWA combination operators outperforms that using the weighted average based on FOAM.

The best overall performance is exhibited by the RiROM system. For the RiROM system, a key step of strategy selection is used. That is, if two ontologies have high label similarity, then the matching process will rely more on linguistic based strategies; while if the two ontologies have high structural similarity, they will employ similarity-propagation based strategies on them [8]. So RiROM can perform better by using the flexible strategy selection.

Our system is outperformed by some systems like coma, falcon and RiROM. The main reason is that, we just use some simple and straight matchers. While in other systems, such as COMA++[5], the graph based matchers are used which have been proved that such kind of matchers can perform well by these systems. From this the lesson we have learned is to integrate some sophisticated base matchers into FOAM.

Although FOAM-OWA is outperformed by coma, falcon and RiROM, there is no big difference between ours and others. For example, the maximal difference between ours and RiROM is 0.058 regarding H(f). Besides, we still have quite a lot space to improve the performance of our system because of the flexibility of OWA operators and the support from the existing and ongoing theoretical and practical study of OWA operators. Furthermore, ours outperform other systems except the top three systems. From Table 1, it can be seen that it is worth to integrate OWA operator into FOAM as the precision and recall are both improved. For example, precision is improved from 0.87 to 0.93 and recall is from 0.78 to 0.80.

5 Conclusion and Future Work

It has been proved that, in most cases, combining the results of multiple matching approaches or matchers is a promising technique to get better results than just using one matcher at a time [1, 3, 2, 10]. In this paper, we integrated OWA aggregation operator into FOAM to provide a novel and promising so-

lution for ontology matching. We summarize the answers for those questions given in Section 2.

1. **Test with more base matchers and complex ontologies**. By testing 23 base matchers and about 50 ontologies, we can see that *At least half* operator outperforms other normal aggregation operators like *average* and *weighted average* in most cases.

2. **Choose different combination operators**. Generally, there are two ways to choose combination operators. One is according to their semantics explained in Section 3.3. Another way is according to various tasks which has been analyzed in our experiments in Section 4.3.1. For example, if a high precision is required, an operator that assigns more weight to the lower similarities may be adequate, e.g. *as many as possible*.

3. **Compare with other matching systems**. Although FOAM-OWA is outperformed by the top three systems coma, falcon and RiROM, there is no big difference (e.g. the maximal difference is 0.058 regarding the harmonic mean of f-measure). As for the flexibility of OWA operators, There is still room for improving the performance of our system. Furthermore, our system outperforms others except the top three ones. We have shown that it is worth to integrate OWA operator into FOAM as the precision and recall are both improved comparing previous version of FOAM.

In the future work, we will extend our current work along two directions. One is the integration of machine learning techniques to obtain OWA weights when rich data sets related to the test ontologies are available. Another direction is combining OWA weights with the weights associated to base matchers using the techniques of the weighted OWA operator [9] when the base matchers have different importance.

References

[1] H. Do and E. Rahm. COMA - a system for flexible combination of schema matching approaches. In *Proc. of VLDB'02*, 2002.

[2] M. Ehrig and S. Staab. QOM - quick ontology mapping. In *Proc. of ISWC'04*, pages 683–697, 2004.

[3] J. Euzenat and P. Valtchev. Similarity-based ontology alignment in OWL-Lite. In *Proc. of ECAI'04*, pages 333–337, 2004.

[4] Q. Ji, W. Liu, G. Qi, and D. Bell. LCS: A linguistic combination system for ontology matching. In *Proc. of KSEM'06*, pages 176–189, 2006.

[5] S. Massmann, D. Engmann, and E. Rahm. COMA++: Results for the ontology alignment contest OAEI 2006. In *Proc. of OM'06*, 2006.

[6] E. Rahm and P. Bernstein. A survey of approaches to automatic schema matching. *The VLDB Journal*, 10(4):334–350, 2001.

[7] P. Shvaiko and J. Euzenat. A survey of schema-based matching approaches. *Journal on Data Semantics IV*, 4(LNCS 3730):146–171, 2005.

[8] J. Tang, Y. Liang, J.Z. Li, and K.H. Wang. Risk minimization based ontology matching. In *Proc. of AWCC'04*, 2004.

[9] V. Torra. The weighted OWA operator. *International Journal of Intelligent Systems*, 12:153–166, 1997.

[10] K. Tu and Y. Yu. Combining mutiple schema-matching strategies based on credibility prediction. In *Proc. of DASFAA'05*, pages 17–20, 2005.

[11] R.R. Yager. On ordered weighted averaging aggregation operators in multicriteria decision making. *IEEE Transactions on Systems, Man and Cybernetics*, 18(1):183–190, 1988.