An Effective Ontology Matching Technique

Ahmed Alasoud, Volker Haarslev, Nematollaah Shiri
Computer Science & Software Engineering, Concordia University
1455 De Maisonneuve West, Montreal, Quebec, Canada
{ahmed_a, haarslev, shiri}@cse.concordia.ca

Abstract. In this paper, we study the ontology matching problem and propose an algorithm, which uses as a backbone a multi-level matching technique and performs a neighbor search to find the correspondences between the entities in the given ontologies. A main feature of this algorithm is the high quality of the matches it finds. Besides, as the result of the initial search introduced, our algorithm converges fast, making it comparable to existing techniques.

1 Introduction

Ontology matching is a fundamental problem in sharing information and integrating ontology sources in numerous applications. We witness a continuous growth in both the number and size of available ontologies. This, on the other hand, has resulted in an increased heterogeneity in the available information. For example, the same entity could be given different names in different ontologies or it could be modeled or described in different ways. The ontology Matching Problem (OMP) is as follows: given ontologies O1 and O2, each describing a collection of discrete entities such as classes, properties, individuals, etc., we want to identify semantic correspondences between the components of these entities. This problem has been the subject of numerous studies, resulting in the development a number of useful and interesting tools and techniques.

Existing matching algorithms often focus on matching a pair of entities at a time, and hardly consider matching *n* entities to *m* entities at the same time, and correspondingly use several similarity measures to solve OMP. We view OMP as an n:m matching problem. Furthermore, to improve matching results, we believe existing methods should be used simultaneously and combined in a multi-level matching framework. This is the subject of our study in this paper. We introduce a neighbor search algorithm, with a proper initialization as an optimization for our multi-level matching algorithm proposed in [1], which improves the matching quality as well as computation time. We have developed a running program and conducted experiments. Our results indicate the proposed neighbor search is effective in improving our multi-level matching algorithm, by finding quality matches efficiently.

In Section 2 we give background definitions. Our search algorithm is introduced in Section 3. An illustrative scenario is provided in Section 4. The experiments and results are presented in Section 5. Section 6 reviews related work, and Section 7 includes concluding remarks and a discussion of future work.

2 Background

In this section, we provide some definitions of concepts and terms used in our work.
Definition 1 (Entity-relationships) Let S be a source ontology and T a target ontology. We use $E^S = \{s_1, s_2, \ldots, s_n\}$ and $E^T = \{t_1, t_2, \ldots, t_m\}$ to denote the set of entities in S and T, respectively. In this work, we limit ourselves to finding mappings for entities of types classes and relationships only.
Definition 2 (Similarity Matrix) This relational matrix, denoted $L(l_{ij})$, includes values in the range [0,1], called the *similarity coefficients*, denoting the degree of similarity between s_i and t_j.
Definition 3 (Matching Matrix) A matching matrix, denoted $Map_{0\text{-}1}$, is a 0-1 matrix with dimension n×m and with entries $r_{ij} \in \{0,1\}$. If $r_{ij} = 1$, it means that s_i and t_j are "matchable." They are unmatchable if $r_{ij} = 0$.
Definition 4 (Matching Space) Matching space includes all possible assignments for the matching matrix, called the *mapping space*. Every assignment is a state in the matching space and represents a solution for the ontology matching problem.

3 A Neighbor Search Algorithm

The proposed neighbor search algorithm has three phases, described in Fig. 1.

```
Algorithm Match(S, T)
begin
/* Initialization phase
     K ← 0 ;
     St0 ←preliminary_matching_techniques(S,T);
     Stf ←St0 ;
/* Neighbor Search phase
     St ←All_Neighbors(Stn);
     While (K++ < Max_iteration) do
/* Evaluation phase
             If score(Stn) > score(Stf) then
                     Stf ←Stn ;
             end if
             Pick the next neighbor Stn ∈ St ;
             St ← St - Stn ;
             If  St = Ø then Return Stf ;
     end
     Return Stf ;
end
```

Fig. 1. The Search Algorithm

First, in the initialization phase, a partial set of similarity measures is applied to the input ontologies to determine a single initial sate St_0 for the search algorithm. In the second phase, we search in the neighborhood of the initial state. The neighbors of state St_0 are the mapping states that can be computed either by adding to or removing from St_0 a couple of vertices, obtained by toggling a bit in the similarity matrix L. So, the total number of the neighbor states will be $n*m$. We evaluate the neighbor states using the following score function v:

$$v = \left(Map_{0-1} \cdot L\right)/k = \sum_{i=1}^{n}\sum_{j=1}^{m} Map_{0-1}(i,j).L(i,j) \Big/ \sum_{i=1}^{n}\sum_{j=1}^{m} Map_{0-1}(i,j) \geq th \ .$$

where $K \geq \min(n,m)$ is the number of matched pairs, n is the number of entities in S, and m is the number of entities in T.

In the third phase (evaluation phase), the algorithm will apply the next level(s) similarity techniques in order to find St_f, the best possible matching state solution.

4 Illustrative Example

Consider simple examples shown in Fig. 2, which are taxonomies for computer ontologies O_1 and O_2.

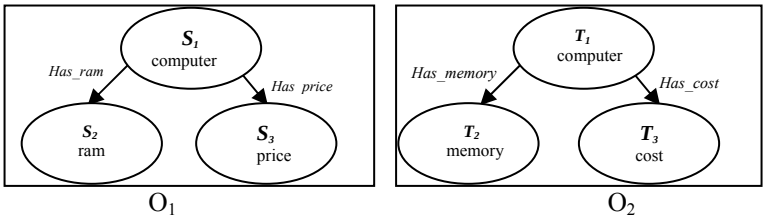

Fig. 2. Computer Ontology Examples

For ease of explanation, we only use three different similarity measures applied in two different phases. There are two similarity measures applied in the first phase to compute the initial state St_0: name similarity (Levenshtein distance) [2] and linguistic similarity (WordNet) [11]. This yields two similarity matrices for the concepts. The first matrix based on name similarity, and the second matrix based on linguistic similarity. Assuming that $th \geq 0.45$, and after normalizing the cost of the two similarity matrices, we get the matrix L. Then L is transformed into the matching matrix $Map_{0\text{-}1}$. Note that we are using $Map_{0\text{-}1}$ and St_n as synonymous.

$$L = \begin{bmatrix} 1.0 & 0.4 & 0.265 \\ 0.463 & 0.534 & 0.083 \\ 0.363 & 0.158 & 0.5 \end{bmatrix} \qquad \text{Map}_{0\text{-}1} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The binary matrix $Map_{0\text{-}1}$ above corresponds to state $St_0=\{(s_1, t_1), (s_2, t_1), (s_2, t_2), (s_3, t_3)\}$, which says entity s_1 is matched to t_1, s_2 is matched to both t_1 and t_2, and s_3 is matched to t_3. Table 1 indicates the binary matrix for other neighboring states together with their score values. In the search phase, 9 neighbors of St_0 will be evaluated to pick the best candidate(s) for the next level. To reduce the cost of the evaluation phase, we filter the neighbor states by keeping $\lceil x\% \rceil$ of the top weighted states for the next level. In phase three, we applied our structure similarity measure proposed in [1]. Finally, the search algorithm will output St_4 which has a highest overall score value, for being structurally more similar.

Table 1. Score value for each state neighbor

Neighbor number	Matched pairs	Score value based on our score function V_{stn}
St_{n1}	$\{(s_2, t_1), (s_2, t_2), (s_3, t_3)\}$	0.499
St_{n2}	$\{(s_1, t_1), (s_1, t_2)\ , (s_2, t_1), (s_2, t_2), (s_3, t_3)\}$	0.5794
St_{n3}	$\{(s_1, t_1), (s_1, t_3)\ , (s_2, t_1), (s_2, t_2), (s_3, t_3)\}$	0.5524
St_{n4}	$\{(s_1, t_1), (s_2, t_2), (s_3, t_3)\}$	**0.678**
St_{n5}	$\{(s_1, t_1), (s_2, t_1), (s_3, t_3)\}$	0.6543
St_{n6}	$\{(s_1, t_1), (s_2, t_1), (s_2, t_2), (s_2, t_3)\ , (s_3, t_3)\}$	0.516
St_{n7}	$\{(s_1, t_1), (s_2, t_1), (s_2, t_2), (s_3, t_1), (s_3, t_3)\}$	0.572
St_{n8}	$\{(s_1, t_1), (s_2, t_1), (s_2, t_2), (s_3, t_2), (s_3, t_3)\}$	0.531
St_{n9}	$\{(s_1, t_1), (s_2, t_1), (s_2, t_2)\}$	0.6656

5 Experiments and Results

Case study (1): In this case study we used the OAEI 2007 benchmark test samples suite [13]. Except for case 206, which is related to French translation, in all other cases we considered, when the *precision* value was less than 1 the *recall* value was equal to 1. We noted all the systems we considered produced all the correct mappings, together with some additional unwanted mappings. The precision of our search algorithm on the other hand we observed did not fall below the *recall* value, i.e., no extra unwanted mappings returned by our framework. However, in test case 206, the reason that the matching result of our search algorithm was not fulfilled was that it did not use translating techniques as one of its underlying techniques. Fig 3. shows the comparison of matching quality of our algorithm and the other 10 systems. To measure a match quality, we have used the following indicators: *precision*, *recall*, and *F-measure*. The version computed here is the harmonic mean of precision and recall [3]. Moreover, Fig. 4 shows an approximate time comparison indicating the scalability of our search algorithm (logarithmic scale). We use MLMA$^+$ to refer to MLMA with the proposed neighbor search algorithm included.

Case study (2): In this case study we used three pairs of ontologies: (1) the MIT bibtex ontology[1] and the UMBC publication ontology[2] which are publicly available, (2) computer ontologies, and (3) ontologies about computer science departments. We have created the second and third pairs of the ontologies. The execution time in seconds for our algorithm over these test cases we measured was, 4.68, 0.547, and 1.719, respectively. A naïve implementation of MLMA would not perform as desired. The MLMA+ is polynomial with respect to the size of the search space $O((|E^S|\times|E^T|)^2)$, where $|E^S|$ is the number of entities in S. All in all, we consider the proposed algorithm as an optimization for MLMA, which we called MLMA$^+$.

[1] http://visus.mit.edu/bibtex/0.1/bibtex.owl.
[2] http://ebiquity.umbc.edu/ontology/publication.owl.

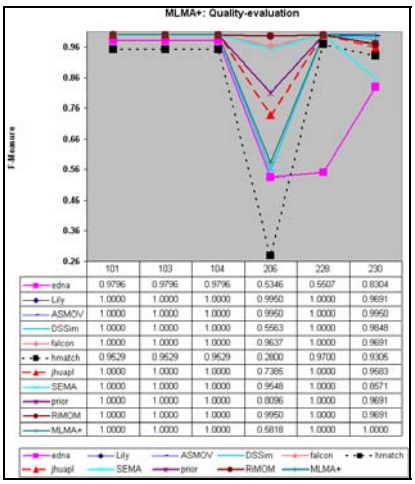

Fig. 3. Quality Comparison

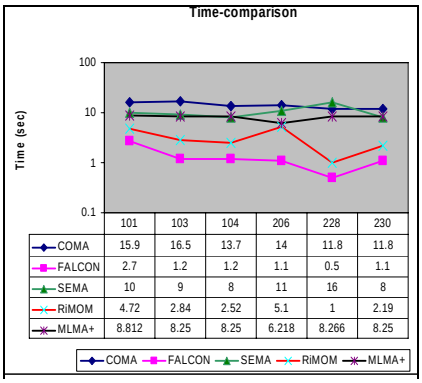

Fig. 4. Efficiency Comparison

6 Related Work

The RiMOM system [9] integrates multiple strategies and applies a strategy selection method to decide the strategy will rely more on it. The proposed method in [12] recommends an alignment strategy for a given alignment problem. The work in [10] has a matching engine which contains diverse libraries that support many match algorithms and strategies. Falcon-AO [8] has two Linguistics matchers and one structural matcher. The results of Falcon-AO were derived either from linguistic or structural matchers. Otherwise, the Falcon-AO results will be generated by combining both matchers with a weighting scheme. Some researchers propose a similarity metric between concepts in OWL ontologies [4] is a weighted combination of similarities of various features in OWL concept definitions. Algorithms such as the one proposed in [7] make use of derived graphs or alternative representations like pair-wise connectivity graphs.

There are three features which make our approach distinct from the aforementioned algorithms and systems. The first is that our matching results are guided by the fact that *n* entities at a time are matched to *m* entities. The second is in the way similarities are transformed into mappings and measured using our multi-match technique in order to deal with a many to many match problem. The third difference is the neighbor search method we introduced for MLMA to improve its efficiency.

7 Conclusions and Future Work

We proposed a neighbor search algorithm, which given an initial mapping state among entities in two ontologies, searches the neighboring states and returns a list of states ranked based on their evaluation scores. We incorporated this search algorithm into our multi-level match algorithm (MLMA) proposed in [1]. This results in MLMA$^+$, a framework for solving ontology match problem, which improves the efficiency of MLMA considerably, due to its use of the neighbor search algorithm. It proceeds by computing an initial state and then performing a search in its neighboring states. We have developed a running prototype of MLMA$^+$ and conducted experiments using some well-known benchmark ontologies. Our results indicated that the proposed search technique improved the overall performance of MLMA. A main characteristic of MLMA$^+$ is its improved efficiency over the basic MLMA obtained through the initial search. We are working on combining the search with machine learning techniques to further improve efficiency and accuracy of MLMA$^+$.

Acknowledgements: This work was supported in part by grants from Natural Sciences and Engineering Research Council (NSERC) of Canada, and by Libyan Ministry of Education.

References

1. Alasoud, A., Haarslev,V., and Shiri, N. A Multi Level Matching Algorithm for Combining Similarity Measures in Ontology Integration, in ODBIS VLDB-Workshop Post-proceedings, LNCS 4623, Springer-Verlag, Berlin, Heidelberg, pp. 1-17, 2007.
2. Cohen, W., Ravikumar, P., Fienberg, S. A Comparison of String Distance Metrics for Name-Matching Tasks. IJCAI-03: 3-78, 2003.
3. Do H.H., Melnik S., and Rahm E. Comparison of schema matching evaluations. In Proc. workshop on Web and Databases, 2002.
4. Euzenat, J. and Valtchev, P. Similarity-based ontology alignment in OWL-Lite. In Proc. 16th European Conference on Artificial Intelligence (ECAI-04), Valencia, Spain, 2004.
5. Euzenat, J., Mochol, M., Shvaiko, P., Stuckenschmidt, H., Svab, O., Svatek, V., Robert, W., Hage, V., and Yatskevich, M. Results of the ontology alignment evaluation initiative 2006. Proc. of ISWC workshop on Ontology Matching, Athens, pages 73–95, 2006.
6. Euzenat, J., Isaac, A., Meilicke, C., Shvaiko, P., Stuckenschmidt, H., Šváb, O., Svátek, V., Robert, W., Hage, V., and Yatskevich, M. Results of the ontology alignment evaluation initiative. Proc. of the ISWC workshop on Ontology Matching, Busan, Korea, Nov. 2007.
7. Hu, W., Jian, N.S., Qu, Y.Z., and Wang, Y.B. GMO: A Graph Matching for Ontologies. In Proc. K-Cap Workshop on Integrating Ontologies, pages 43-50, 2005.
8. Hu, W., Cheng,G., Zheng, D., Zhong, X., and Qu, Y. The results of Falcon-AO. In Proc. Int'l workshop on Ontology Matching (OM), Athens, Georgia, U.S.A, Nov 5, 2007.
9. Li, Y., Li, J., Zhang, D., and Tang, J. Results of ontology alignment with RiMOM. In Proc. Int'l workshop on Ontology Matching (OM), Athens, Georgia, U.S.A, Nov 5, 2007.
10. Massmann, S., Engmann, D., and Rahm, E., and Tang, J. Results of ontology alignment with COMA++. In Proc. Int'l workshop on Ontology Matching (OM), U.S.A, Nov 5, 2006.
11. Pedersen, T., Patwardhan, S., Patwardhan, S. WordNet::Similarity – Measuring the Relatedness of Concepts. In Proc. of 19th National Conf.on AI, San Jose, CA, 2004.

12. Tan, H., Lambrix, P. A method for recommending ontology alignment strategies. In Proceedings of the 6th International Semantic Web Conference, Busan, Korea, 2007.
13. http://oaei.ontologymatching.org/2007/benchmarks/