Exploring Pattern Mining for Solving the Ontology Matching Problem

Hiba Belhadi[1],Karima Akli-Astouati [1] ,Youcef Djenouri[2] ,and Jerry Chun-Wei Lin [3] ,

[1] Department of Computer Science, USTHB, Algiers, Algeria.
{hbelhadi,kakli}@usthb.dz
[2] Department of Computer and Information Sciences, NTNU, Trondheim, Norway.
youcef.djenouri@ntnu.no
[3] Western Norway University of Applied Sciences, Bergen, Norway.
jerrylin@ieee.org

Abstract. This paper deals with the ontology matching problem, and proposes a pattern mining approach that exploits the different correlation and dependencies between the different properties to select the most appropriate features for the matching process. The proposed method first discovers the frequent patterns from the ontology database, and then find out the most relevant features using the patterns derived. To demonstrate the usefulness of the suggested method, several experiments have been carried out on the OAEI (Ontology Alignment Evaluation Initiative) and DBpedia ontology databases. The results show that our proposal outperforms the state-of-the-art ontology matching approaches in terms of both execution time and quality of the matching process.

Keywords: semantic web · ontology matching· matching instances · frequent patterns

1 Introduction

The ontology matching is the process to build a bridge between different instances that represent the same real world, by the set of instances, where each instance is characterized by different properties. It is applied in diverse fields such as biomedical data [15], and Natural Language Processing [9]. Trivial solutions for ontology matching compares each instance of the first ontology with each instance of the second ontology by taking into account all the properties of both ontologies. $n \times n' \times m \times m'$ comparisons are required to find the alignment, where n and n' are the numbers of instances, and m and m' are the corresponding numbers of the data properties of the first ontology and the second ontology, respectively. These solutions have polynomial complexity. However, for some high dimensional data like DBpedia ontology [4] containing 4,233,000 instances, and 2,795 different properties, 144×10^{18} possible comparisons are required for such

[4] http://wiki.dbpedia.org/Datasets

solutions. As a result, the matching process became a high time consuming, and also it can be reduced on the alignments quality performance.

Emergent solutions to the ontology matching problem attempt to improve the quality of the overall matching process, by exploiting the enumeration search space using partitioning algorithms [3], evolutionary algorithms [18] and using high performance computing [16]. However, the overall performance of these algorithms is still low when dealing with high dimensional data. Pattern mining is a data mining technique that finds frequently co-occurring items in a database, and accordingly provides relevant patterns useful in the decision making process. Pattern mining largely applies as preprocessing step for solving complex problems [5, 4]. Motivated by the success of the pattern mining discovery process, and in order to improve the overall performance of the ontology matching problem on high dimensional data, this paper investigates the use of pattern mining in selecting the most relevant features for solving the ontology matching based instance problem. In the pattern mining related literature, several algorithms have been proposed such as the Apriori [1], Fpgrowth [8], and many others. These approaches are both time-consuming and memory-consuming, especially when dealing with a low minimum support value. Recently, SSFIM [7] was proposed to extract the frequent itemsets using only a single pass, and it was proven to be non-sensitive to the minimum support value. The experimental study reported in [6] reveals that the SSFIM outperforms the state-of-the-art pattern mining algorithms. Therefore, in this work, the SSFIM is adopted to study the correlations of the properties of the given ontologies.

The main contributions of our work are threefold:

- Propose a new framework called PMOM (Pattern Mining for Ontology Matching) which adopts the pattern mining techniques to solve the ontology matching problem. In this context, SSFIM [7] is applied to discover the frequent patterns of the given ontologies.
- Develop a new strategy based on the extracted patterns for selecting the most relevant properties of the given ontologies. This is realized by computing the probability of each property in the set of the derived patterns.
- An intensive experiments have been performed to demonstrate the usefulness of the PMOM framework. The results reveal that PMOM outperforms the state-of-the art ontology matching algorithms.

The rest of this paper is organized as follows. Section 2 reviews some existing works related to the ontology matching problem. Section 3 introduces the ontology matching based instance problem. Section 4 presents a detail explanation of the PMOM framework. The evaluation performance is sketched in Section 5. Section 6, concludes this paper with some perspectives for a future work.

2 Related Work

In the last decade, several works have been proposed for solving the ontology matching problem [14, 12]. Wang et al. [10] developed a generic VMI approach,

which reduces the number of similarity computations by introducing multiple indexes, and candidate selection rules. This approach suffers from the quality when increasing with the number of instances. Li et al. [17] proposed an approach based on the hypothesis that two entities of the same real-world object could be matched if they are linked to some previously matched entities. It first combines multiple lexical matching strategies using a novel voting-based aggregation method, it then utilizes the structural information and the correspondences already found to discover additional ones. Hu et al. [13] proposed RiMOM presented in the OAEI 2013 competition. It represents an iterative matching framework where the distinctive information is based on a blocking strategy to reduce the number of candidate instance pairs. It uses predicates and their distinctive object feature as a key of the index for the instances. It also employs a weighted exponential function based similarity aggregation approach to assure the high accuracy of instance matching. Niu et al. [11] developed a EIFPS (Extended Inverse Functional Property Suite) approach, a semi-supervised learning algorithm, to recursively refine the matching process by using rules extracted by the association rule mining approach. A small number of existing properties are used as seeds and the matching rules are treated as parameters for maximizing the precision. Sergio et al. [2] proposed LOM (Learning Objects Metadata) by presenting the power of homogeneity resources in e-learning context. The application of an original associative classifier to the problem of ontology matching is investigated, in order to extend and improve the available tools for online learning in the semantic way. The approach uses a feature based similarity function that requires previous knowledge of the training set. Although, the ontology matching-based approaches perform well on small and medium ontology databases, they are inefficient, in terms of runtime performance and solution's quality, for large ontologies (i,e high number of instances), and high dimensional data (instances with high number of properties). To deal with these two challenging issues, in this work, we present a pattern mining-based approach that explores the discovered patterns to select the most relevant features for solving the ontology matching process. Before detailing on our proposal, the next section presents the ontology matching based instance problem.

3 Ontology Matching Based Instance Problem

The goal of the ontology matching problem is to determine the common features between two ontologies. The result of this process is to represent the alignment between these ontologies. For that, an ontology O described by a set of instances $I = \{I_1....I_k\}$. Each of which is defined by a set of attributes (data properties) $P = \{p_1....p_n\}$. The properties may have more than one value.

It exists many variants of ontology matching problem. We are interesting in this work to ontology matching based instance. This variant of matching considers the common instances between two ontologies, with considering that the instance could have the same values for some properties and could also have missing values for other properties. Note that the name of properties for each

ontology can be different, this issue causes the alignment more difficult. Thus, all the values of the instances of the first ontology should be scanned and compared to all the values of the instances of the second ontology. Consequently, the aim of the ontology matching based instance problem is to find the same information represented differently.

The alignment's result depends to the method used in the matching process, for that each matching can result a different number of instances in alignment. For that, each resulted alignment is evaluated and compared to an alignment reference. Alignment reference represents an alignment that is suggested by the domain expert. The alignment reference contains all the common instances between the ontologies.

Fig. 1 represents a sketch of the ontology matching based instance problem. Consider two ontologies, O_1 and O_2, the first step aims to convert each ontology represented by the set of concepts and terms into a matrix, where each element $[i, j]$ represents the value of the j^{th} property in the i^{th} instance. The matching process is then performed to derive an alignment between these two ontologies. The alignment reference represents the set of instances in common into two ontologies. The optimal matching between O_1 and O_2 is $i_{11} = i_{21}$ and $i_{14} = i_{22}$. For that, we can conclude that the property P_{11} and P_{23} are equal, respectively for P_{12} and P_{22}, and for P_{13} and P_{21}.

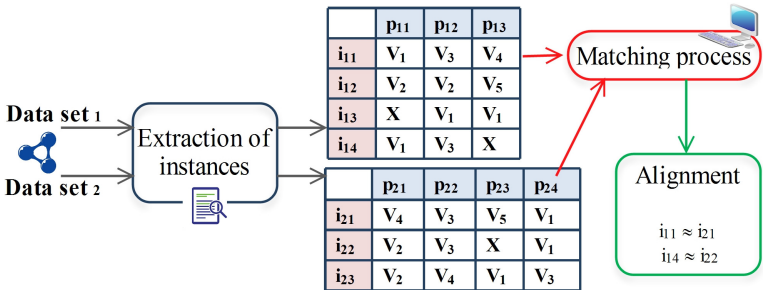

Fig. 1. Ontology Matching based Instance Illustration.

4 PMOM: Pattern Mining for Ontology Matching based instances

4.1 Overall Framework

In this part, we present the main components of the proposed framework called PMOM (Pattern Mining for Ontology Matching based instances). The aim of PMOM is to improve the ontology matching based instance problem by taking into account the relevant features of the two ontologies to be aligned. This reducing allows on the one hand to boost the matching process for finding the

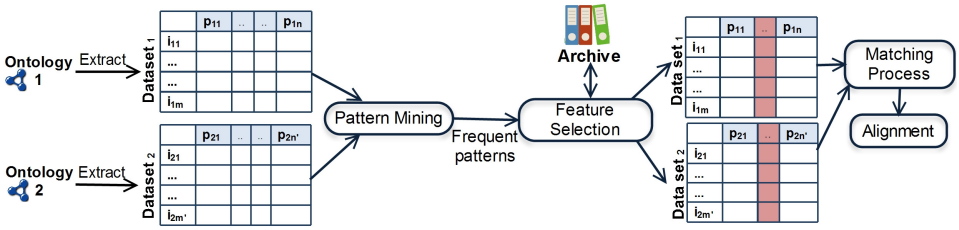

Fig. 2. PMOM Framework

common instances between two ontologies, on the other hand, it aims to improve the quality of the resulted alignment.

PMOM is mainly composed into two steps: feature selection and matching process steps (See Fig. 2 for more details). The feature selection step is first performed to the set of attributes for each ontology, which results in an optimal subset of attributes that represents perfectly the two ontologies. This step is considered as a pre-processing step, (it will be executed only one time). To do so, an archive folder will be constructed for each ontology in the ontology base system. The matching process is then applied between the instances of the ontologies by taking only the attributes selected of the above step. The K-cross-validation model is used here, where at each pass of the algorithm, the training and the test alignments are performed. For the training matching process, the proposed model is learned to fix the best parameters. If the alignment rate exceeds the given threshold, then the test alignment is started. In this work, we are interested in the feature selection part, by proposing the pattern mining strategy for selecting the relevant attributes for the input ontologies. Any existing methods could be used for the ontology matching process.

4.2 Feature Selection

This strategy studies the correlation between the data properties of the ontologies to select the best features of the matching process. It is inspired by the pattern mining process [1] which is used to extract the most relevant data properties that cover the maximum number of possible instances. PMOM denotes the extraction of the relevant patterns that satisfies the minimum support constraint (*minsup*) from the transactions. Motivated by the success of SSFIM [6] in discovering the pattern mining problem, this work proposes the use of SSFIM in PMOM framework. The mining process is performed through two main steps: generation and extraction. In the generation step, starting from the first instance I_1, we refer $Pattern(I_1)$ by the set of all possible combinations of the literals of this instance. The result is added to the hash table H by creating an entry for each itemset in $Pattern(I_1)$. The frequency of each pattern is initialized by one in the hash table H. Then, the patterns of the second instance I_2 are generated for each pattern in $Pattern(I_2)$; if this pattern exists in H, then, its frequency is incremented by one; otherwise, a new entry is created with the frequency set to

Algorithm 1 Feature Selection Step

```
Input: I : The set of instances
P: The set of all data properties
S: The set of frequent literals
σ: Minimum support threshold
μ: Interestingness threshold
Output: SP : The set of selected data properties
for each instance e ∈ I do
    F_e ← Pattern(e)
    for each element i ∈ F_e do
        if i ∈ H then
            Freq_i++
        else
            AddH(i,1)
        end if
    end for
end for
S ← ∅
for each element h ∈ H do
    if Support(h) > σ then
        S ∪ h
    end if
end for
SP ← ∅
for each property i ∈ P do
    if Probability(i, S) > μ then
        SP ∪ i
    end if
end for
return SP
```

1. This process is repeated until all instances I are processed. The second step aims to extract the frequent patterns (frequent literals in our case) from the hash table H. For that purpose, the support of each pattern t is computed (See EQ 1); if the support of t is greater than $minsup$, then t is called the frequent literal and added to the set of frequent literals S.

$$Support(t) = \frac{h(t).freq}{|I|} \tag{1}$$

Based on the frequent literals S discovered above, the appropriate set SP is selected. The probability $P(i, S)$ denotes the probability of the apparition of the i^{th} property in the set of frequent literals S. A threshold μ which is between [0-1], is used to select the appropriate data properties. Indeed, for each property, if its probability value is greater than μ then, it is added to SP set. The algorithm of this strategy is given in Algorithm 1.

4.3 Matching Process

After the selection of the appropriate properties, it is time to compare the instances of the basic ontology to the second one. In this part, we consider two ontologies, the basic ontology BO that will be matched with the second ontology O. $< P, I >$ is the set of property P and the instances I of the basic ontology. $< P', I' >$ is the set of propertyP' and the instances I' of the second ontology. P and P' are the set resulted by the feature selection described above. The iterative

Algorithm 2 Matching Process

```
1:  Input: I: Set of instance of BO
2:  I': Set of instance of O
3:  P: Set of selected property of BO
4:  P': Set of selected property of O
5:  Output: AL: Alignment set
6:  for each instance i ∈ I do
7:      P ← SetProperties(i)
8:      for each instance j ∈ I' do
9:          P' ← SetProperties(j)
10:         L ← ∅
11:         for each property p ∈ P do
12:             for each property p' ∈ P' do
13:                 if Value(p,p') then
14:                     L ∪ {p,p'}
15:                 end if
16:             end for
17:         end for
18:         if L !=∅ then
19:             AL ∪ ({ID_i,ID_j}∪L)
20:         end if
21:     end for
22: end for
23: return AL
```

matching consists to scan the whole set of instances I of the basic ontology, and compares it with each instance in the set of instances I' of the second ontology. The comparison between two instances is established by checking each value in the i^{th} instance in BO with all the values in the j^{th} instance in O.

From the line 6 to line 22, Algorithm 2 scans the whole instances I of BO and from the line 8 to line 21, it scans the instances I' of O. The function $SetProperties()$ recuperates the properties of both i^{th} and j^{th} instances respectively. From the line 13 to line 15, it compares the value of the properties p and p' of the i^{th} and the j^{th} instances, respectively. If they are the same, they will be added to the set L. Finally, if the set L is not empty, then the i^{th} and j^{th} instances are added to alignment set AL.

5 Performance and evaluation

To validate the usefulness of the PMOM framework, extensive experiments have been carried out using well-known ontology matching databases. The approach has been implemented in Java [5] and experiments have been run on a desktop machine equipped with an Intel $I7$ processor and $16GB$ memory. Two sets of well-known ontology databases that are often used by the ontology matching community are considered in these experiments:

1. OAEI databases (Ontology Alignment Evaluation Initiative): These databases are retrieved from [6]. Table 1 lists the number of instances and properties for these ontology databases.

[5] Code available at https://github.com/YousIA/PMOM
[6] http://oaei.ontologymatching.org

Table 1. OAEI Databases Description

Ontology Name	#Instances	#properties
OntoA_dis	29,645	11
OntoB_dis	15,556	11
OntoA_rec	15,556	11
OntoB_dis	1,708	11
Onto_a_id	1,330	5
Onto_b_id	2,649	4
Onto_a_sim	173	5
Onto_b_sim	172	5
*onto*101	57	46
*onto*104	56	46
*onto*202	57	46
*onto*230	47	49
$IIMB$000	12,333	13
$IIMB$104	12,338	13
*person*11	500	14
*person*12	500	12
*person*21	600	14
*person*22	400	12

2. DBpedia[7]: It is a hub data that represent the Wikipedia knowledge and make this structured information available on the Web. This database ontology contains 4,233,000 instances, and 2,795 different properties.

The solution's quality of the ontology matching process is evaluated using Precision, Recall and Fmeasure as follows:

1. **Precision** Given a reference alignment R, the precision of some alignment A is a function P : A × A → [0..1] such that:

$$Precision(A, R) = \frac{|R \cap A|}{|A|} \tag{2}$$

2. **Recall** Given a reference alignment R, the recall of some alignment A is a function R : A × A →[0..1] such that:

$$Recall(A, R) = \frac{|R \cap A|}{|R|} \tag{3}$$

3. **Fmeasure** It combines the precision and recall measures as follows:

$$Fmeasure(A, R) = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{4}$$

The minimum support value of PMOM is first tuned using OAEI databases. The best configuration of PMOM is then compared with state-of-the art ontology matching based instance algorithms using DBpedia database.

[7] http://wiki.dbpedia.org/Datasets

Table 2. Comparison of Recall, Precision, and Fmeasure of PMOM, the EIFPS, and the RiMOM using the DBpedia by varying both the percentage of instances (%I) and the percentage of the data properties (%P) from 20% to 100%

% I	%P	PMOM			EIFPS			RiMOM		
		Rec.	Prec.	Fmeas.	Rec.	Prec.	Fmeas.	Rec.	Prec.	Fmeas.
20	20	0.97	0.95	0.96	0.97	0.94	0.95	**0.98**	**0.95**	**0.96**
	50	**0.97**	**0.95**	**0.96**	0.92	0.92	0.92	0.95	0.93	0.94
	80	**0.97**	**0.95**	**0.96**	0.90	0.91	0.90	0.93	0.92	0.92
	100	**0.97**	**0.95**	**0.96**	0.88	0.90	0.89	0.89	0.90	0.89
	20	**0.96**	**0.94**	**0.95**	0.93	0.92	0.92	0.95	0.92	0.93
	50	**0.96**	**0.94**	**0.95**	0.89	0.87	0.88	0.91	0.89	0.90
	80	**0.96**	**0.94**	**0.95**	0.87	0.84	0.85	0.89	0.86	0.87
	100	**0.96**	**0.94**	**0.95**	0.85	0.82	0.83	0.87	0.83	0.85
80	20	**0.95**	**0.92**	**0.93**	0.90	0.89	0.89	0.91	0.90	0.90
	50	**0.95**	**0.92**	**0.93**	0.88	0.86	0.87	0.89	0.88	0.88
	80	**0.95**	**0.92**	**0.93**	0.82	0.80	0.81	0.83	0.81	0.82
	100	**0.95**	**0.92**	**0.93**	0.78	0.75	0.76	0.80	0.79	0.79
100	20	**0.94**	**0.90**	**0.92**	0.85	0.82	0.83	0.87	0.86	0.86
	50	**0.94**	**0.90**	**0.92**	0.83	0.81	0.82	0.84	0.82	0.83
	80	**0.94**	**0.90**	**0.92**	0.78	0.75	0.76	0.80	0.77	0.78
	100	**0.94**	**0.90**	**0.92**	0.74	0.70	0.72	0.73	0.72	0.72

5.1 Parameter Settings

Extensive tests have been carried out to empirically tune the minimum support value of the proposed PMOM using *Person* database available in OAEI. This database consists of two ontologies Person1 and Person2, where the original record is modified by adding property values to generate other record. In Person1, the difference between the two records is one modification; while Person2 a maximum of 3 modifications by record, and maximum 10 duplications of the original record. Person1 contain 500 instances in the two files (Person11 and Person12) with 400 instances in the reference alignment, while Person2 contains 600 instances in Person21 and 400 instances in Person22, with 400 instances in the reference alignment. At each test, the Fmeasure value is determined. By varying the minimum support threshold from 1% to 100%, the optimal value of PMOM was 30%. Thus, this best value is used for the remaining of the experiments.

5.2 Comparison of the PMOM and State-of-the-art Ontology Matching Algorithms

The experiment aims to compare PMOM with the state-of-the-art algorithms (EIFPS [11], and the RiMOM[13]) using the DBPedia ontology database. Fig. 3 shows the runtime of the three approaches at the percentage of data properties from 20% to 100%, considering all instances. When the number of matching varied from 100 to 1,000,000, PMOM outperforms the two other approaches.

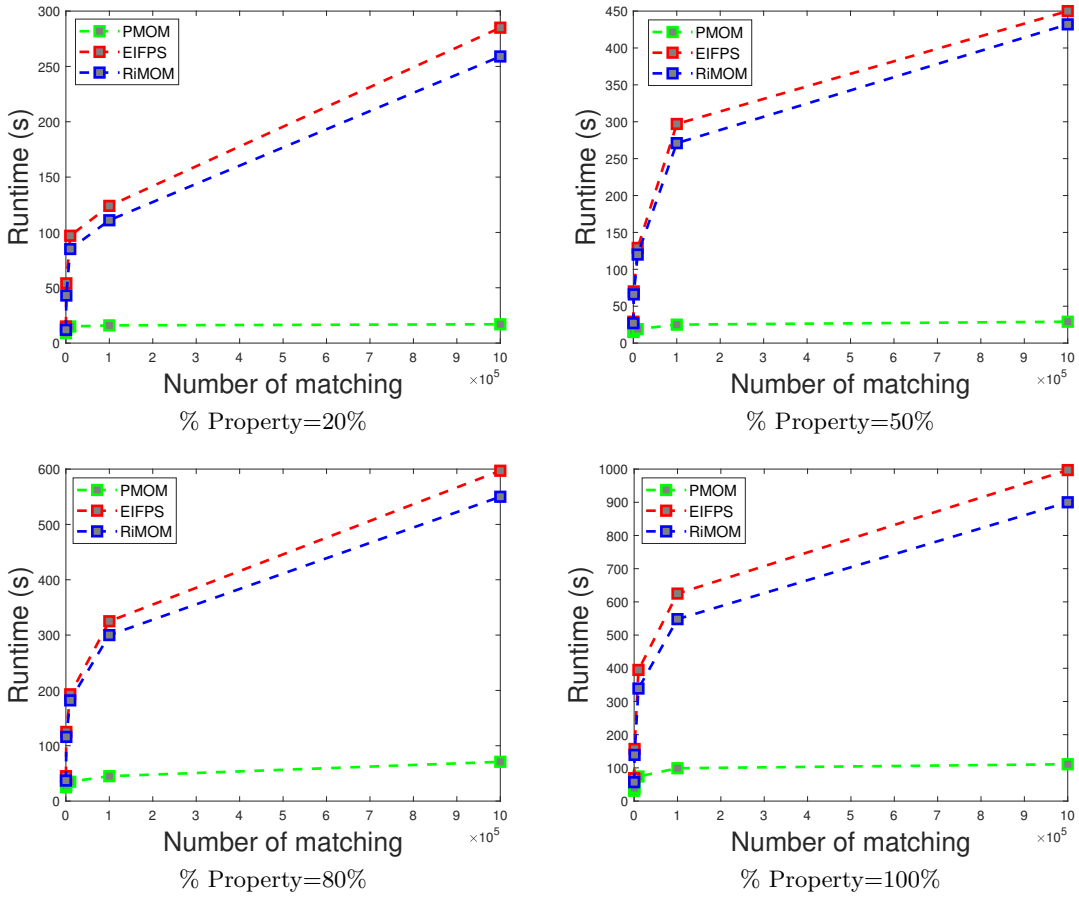

Fig. 3. Comparison of the runtime performance (in seconds) of the PMOM, the EIFPS, and the RiMOM using the DBpedia by varying the percentage of the data properties (%P) from 20% to 100%

Moreover, the runtime of PMOM stabilized at a high number of data properties, where the two baseline approaches were highly time-consuming for a large number of instances and a large number of matchings. Thus, EIFPS and RiMOM need more than 900 seconds for dealing 1,000,000 matching in the whole DBpedia ontology database, whereas, it took only 111 seconds for PMOM. These results were obtained using the preprocessing step, where only the most relevant features were selected using the pattern mining approach.

Table 2 compares the quality of matching of PMOM framework and the baseline algorithms (the EIFPS and the RiMOM) using the DBpedia ontology database. By varying the percentage of data properties and the percentage of instances from 20% to 100%, PMOM outperforms the other two algorithms re-

garding the quality (recall, precision, and fmeasure) in all cases, except in the first case that containing 20% of data properties and instances. Moreover, the results showed that the quality of PMOM is non-sensitive to the number of data properties and the number of instances. Thus, the quality of PMOM exceeds 92%, whereas the quality of the EIFS and RiMOM does not reach 70% and 72%, respectively. These results were obtained thanks to the feature selection procedure, which find out the most relevant data properties of the given ontologies.

6 Conclusions

This paper presents PMOM (Pattern Mining for Ontology Matching based instances) framework, for ontology matching problem. The approach explores different correlations between the data properties and selects the most frequent data properties describing the overall instances of the given ontology. To evaluate PMOM framework, intensive experiments have been carried on OAEI ontology databases and DBpedia database. The results show that PMOM outperforms the baseline methods (EIFPS, and RiMOM) in terms of execution time and solution's quality. As future work, we plan to explore other data mining techniques for ontology matching problem. Dealing with big ontology databases is also in our future agenda.

Acknowledgment

Youcef Djenouri's work was carried out at the Norwegian University of Science and Technology (NTNU), funded by a postdoctoral fellowship from the European Research Consortium for Informatics and Mathematics (ERCIM).

References

1. Agrawal, R., Imieliński, T., Swami, A.: Mining association rules between sets of items in large databases. In: Acm sigmod record. vol. 22, pp. 207–216. ACM (1993)
2. Cerón-Figueroa, S., López-Yáñez, I., Alhalabi, W., Camacho-Nieto, O., Villuendas-Rey, Y., Aldape-Pérez, M., Yáñez-Márquez, C.: Instance-based ontology matching for e-learning material using an associative pattern classifier. Computers in Human Behavior **69**, 218–225 (2017)
3. Del Vescovo, C., Parsia, B., Sattler, U., Schneider, T.: The modular structure of an ontology: Atomic decomposition. In: IJCAI Proceedings-International Joint Conference on Artificial Intelligence. vol. 22, p. 2232 (2011)
4. Djenouri, Y., Djamel, D., Djenoouri, Z.: Data-mining-based decomposition for solving maxsat problem: Towards a new approach. IEEE Intelligent Systems (2017)
5. Djenouri, Y., Belhadi, A., Fournier-Viger, P., Lin, J.C.W.: Fast and effective cluster-based information retrieval using frequent closed itemsets. Information Sciences **453**, 154–167 (2018)

6. Djenouri, Y., Comuzzi, M., Djenouri, D.: Ss-fim: Single scan for frequent itemsets mining in transactional databases. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining. pp. 644–654. Springer (2017)
7. Djenouri, Y., Djenouri, D., Lin, J.C.W., Belhadi, A.: Frequent itemset mining in big data with effective single scan algorithms. Ieee Access **6**, 68013–68026 (2018)
8. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: ACM sigmod record. vol. 29, pp. 1–12. ACM (2000)
9. Iwata, T., Kanagawa, M., Hirao, T., Fukumizu, K.: Unsupervised group matching with application to cross-lingual topic matching without alignment information. Data mining and knowledge discovery **31**(2), 350–370 (2017)
10. Li, J., Wang, Z., Zhang, X., Tang, J.: Large scale instance matching via multiple indexes and candidate selection. Knowledge-Based Systems **50**, 112–120 (2013)
11. Niu, X., Rong, S., Wang, H., Yu, Y.: An effective rule miner for instance matching in a web of data. In: Proceedings of the 21st ACM international conference on Information and knowledge management. pp. 1085–1094. ACM (2012)
12. Otero-Cerdeira, L., Rodríguez-Martínez, F.J., Gómez-Rodríguez, A.: Ontology matching: A literature review. Expert Systems with Applications **42**(2), 949–971 (2015)
13. Shao, C., Hu, L.M., Li, J.Z., Wang, Z.C., Chung, T., Xia, J.B.: Rimom-im: a novel iterative framework for instance matching. Journal of computer science and technology **31**(1), 185–197 (2016)
14. Shvaiko, P., Euzenat, J.: Ontology matching: state of the art and future challenges. IEEE Transactions on knowledge and data engineering **25**(1), 158–176 (2013)
15. Smith, B., Ashburner, M., Rosse, C., Bard, J., Bug, W., Ceusters, W., Goldberg, L.J., Eilbeck, K., Ireland, A., Mungall, C.J., et al.: The obo foundry: coordinated evolution of ontologies to support biomedical data integration. Nature biotechnology **25**(11), 1251 (2007)
16. Thayasivam, U., Doshi, P.: Speeding up batch alignment of large ontologies using mapreduce. In: Semantic Computing (ICSC), 2013 IEEE Seventh International Conference on. pp. 110–113. IEEE (2013)
17. Wang, Z., Li, J., Zhao, Y., Setchi, R., Tang, J.: A unified approach to matching semantic data on the web. Knowledge-Based Systems **39**, 173–184 (2013)
18. Xue, X., Liu, J.: Collaborative ontology matching based on compact interactive evolutionary algorithm. Knowledge-Based Systems **137**, 94–103 (2017)