A Linear Program For Holistic Matching : Assessment on Schema Matching Benchmark

Alain Berro* , Imen Megdiche* , Olivier Teste*

* IRIT UMR 5505, University of Toulouse, CNRS, INPT,
UPS, UT1, UT2J, 31062 TOULOUSE Cedex 9
{Berro,Megdiche,Teste}@irit.fr

Abstract. Schema matching is a key task in several applications such as data integration and ontology engineering. All application fields require the matching of several schemes also known as "holistic matching", but the difficulty of the problem spawned much more attention to pairwise schema matching rather than the latter. In this paper, we propose a new approach for holistic matching. We suggest modelling the problem with some techniques borrowed from the combinatorial optimization field. We propose a linear program, named LP4HM, which extends the maximum-weighted graph matching problem with different linear constraints. The latter encompass matching setup constraints, especially cardinality and threshold constraints; and schema structural constraints, especially superclass/subclass and coherence constraints. The matching quality of LP4HM is evaluated on a recent benchmark dedicated to assessing schema matching tools. Experimentations show competitive results compared to other tools, in particular for recall and HSR quality measures.

Keywords: Schema Matching, Linear Programming, Holistic Matching, Matching Quality Assessment

1 Introduction

Schema matching problem is among the most studied problems in the literature. It represents a key task in several application fields such as data integration, ontology engineering, web services composition, query answering in the web, and so on [10]. The schema matching task consists of identifying correspondences, also named mappings or alignments, between schemes [18]. Almost all the approaches transform inputted schemes into an internal data representation, such as graphs or graph-like representations [1] [19]. Various forms of schemes cited in [10] such as dictionaries, taxonomies, XML/DTD, relational databases, can be internally transformed into graph-like representation (trees, forests, rooted directed acyclic graphs, etc), which have hierarchical organization of nodes.

Furthermore, the number of input schemes declines two types of matching approaches as reviewed in [18]: pairwise matching for two input schemes and holistic matching for several input schemes. In the scientific literature, pairwise schema matching gained much more attention than holistic schema matching. This is due to the different challenges [21] and difficulties of the matching task. However, the availability and the rapid evolution of a huge variety of data requires the integration of several data sources at the same time. Even if, some incremental solutions [5] have been proposed to deal with several or large data sources, the incremental process suffers from the closure of the solution.

In this paper, we define a new holistic approach for matching schemes having a graph-like structure. This approach also allows a holistic matching of the schemes of open data tables [4] represented as an hierarchy of concepts in the works of [3].

The approach combines combinatorial optimization techniques and the characteristics of schema matching problem. It consists of an extension of the maximum-weighted graph matching problem [20] adapted with different constraints related to the schema matching problem. The constraints are related to the mapping cardinalities, the threshold setup, super/subclass relations and structural coherence. The main contributions of our approach are as follows:

- it returns a global optimal solution as it extends the maximum-weighted graph matching problem [10];
- it can be used without setting threshold which makes a gap compared to the other tools;
- it is applicable for pairwise and holistic matching with a theoretical polynomial time [20].

Our approach is evaluated on the recent benchmark proposed by [6]. This benchmark is devoted to assess the quality of matching tools. Hence, the matching quality of our approach is confronted to the matching quality of different referenced solutions.

The remainder of this paper is devoted to the description of our approach in section 3, then we present the experimental assessments of our approach in section 4 and we conclude in section 5. In the following section, we briefly describe related works.

2 Related works

A schema matching task can be summed up by the general workflow described by [18]. This task is composed of a pre-processing step, an execution step for one or several matchers, which can be element-level matchers or structural-level matchers [22] then a combination of matcher(s) results and finally a selection of mappings.

We can notice according to [9] [10] that different pairwise approaches, especially in the field of ontology matching, have reduced one or several steps in the workflow described above into a combinatorial optimization problem. S-Match [12] reduces the semantic matching to the propositional validity problem, which is theoretically a co-NP hard problem. The elements of schemes are translated into logical formulas and the semantic matching consists of resolving propositional formula constructed between elements. Similarity Flooding (SF) [16] reduces the selection of the mappings to the stable marriage problem, which returns a local optimal solution [10]. The SF approach proposes a structural-matcher which propagates similarities between neighbourhood nodes until a fixed point computation. OLA [11] reduced the selection of mappings into a weighted bipartite graph matching problem. This approach models structural similarity computation as a set of equations of the different properties of ontologies. The pairwise matcher CODI [13] implements the probabilistic markov logical framework presented in [17]. This approach transforms the matching problem into a maximum-a-posteriori (MAP) optimization problem which is equivalent to Max-Sat problem (NP-hard).

Furthermore, we highlight the existence of some approaches which have been proposed as a platform for the matching task. For instance COMA++ [2] and YAM [6]. COMA++ (COmbination of Matching Algorithms) is a generic matcher offering to users several strategies for matching schemes and several features to combine and reuse the results of matchings. YAM is a schema matcher factory. It uses machine learning techniques to tune thresholds and other parameters to propose a suitable matcher according to the evaluated dataset.

Discussion. Our solution LP4HM and CODI perform, at the same time, the structural matching phase without additional structural similarity computation and the mapping extraction phase. CODI is based on a pairwise approach whilst LP4HM is a holistic approach. For pairwise ontology matching, CODI is more generic than LP4HM because it considers all type of properties. Whereas, LP4HM considers only hierarchical relationships expressed with the subclass property in the ontologies. The integer linear program (ILP) of LP4HM is reduced, in pairwise scenarios, to the maximum-weighted bipartite graph matching problem just like what OLA has done for the extraction phase. Compared to OLA we do not compute structural similarities but we encoded structural properties as linear constraints. The complexity of the maximum-weighted bipartite graph matching problem with ILP is polynomial [20] even with the simplex algorithm [20]. For holistic scenarios, our ILP is reduced to the maximum-weighted non-bipartite graph matching problem [20] which can be also solved in polynomial time [8]. Unlike CODI whose pairwise approach is reduced to a NP-Hard problem, our proposed solution extends a polynomial problem in both pairwise and holistic versions. YAM have to be run several times to learn a threshold whilst LP4HM uses a predefined threshold and it generates an optimal solution with a unique run. Moreover, using a threshold in our approach is an optional feature.

3 A Linear Program For Holistic Matching : LP4HM

The idea of the LP4HM matcher is to extend the maximum-weighted graph matching (MWGM) problem with additional linear constraints. The authors of [20] define the MWGM problem for a graph G as follows: ”to find a matching (= set of disjoint edges) M in G with a weight $w(M)$ as large as possible”. It is clear that the definition of the graph matching problem in combinatorial optimization field is different from the definition of the schema matching problem. But a reduction holds between both problems. Indeed, we can consider that G is composed of (i) the nodes representing schema elements and (ii) edges representing virtual connections between these nodes and having as weights the similarities between the concepts of the nodes. From this reduction, searching 1:1 correspondences (i.e a node matches at most with only another node) for the schema matching problem is the same as finding a set of disjoint edges with a maximum weight in G. Moreover, pairwise and holistic schema matching problems correspond to bipartite and non-bipartite MWGM problem [20]. The latter have been proved to be theoretically polynomial.

To illustrate these reductions and our linear program, we give the running example of Fig.1. This example represents a part of the schemes of open data tables, which have been transformed into a graph-like structure by the application of the approach of [3]. The data are available on the following links[1]. Each graph in Fig.1 has continuous edges representing their structure. Between each pair of graph, we have dotted edges having as weight similarities. If we consider a graph G composed of the nodes of G_1 and G_2 and the dotted weighted edges between them, resolving pairwise matching between G_1 and G_2 is equivalent to resolving the MWGM problem in the bipartite graph G. If we consider a graph G composed of the nodes of the three graphs G_1, G_2 and G_3 and the dotted edges, resolving holistic matching between G_1, G_2 and G_3 is equivalent to resolving a MWGM in the non-bipartite graph G.

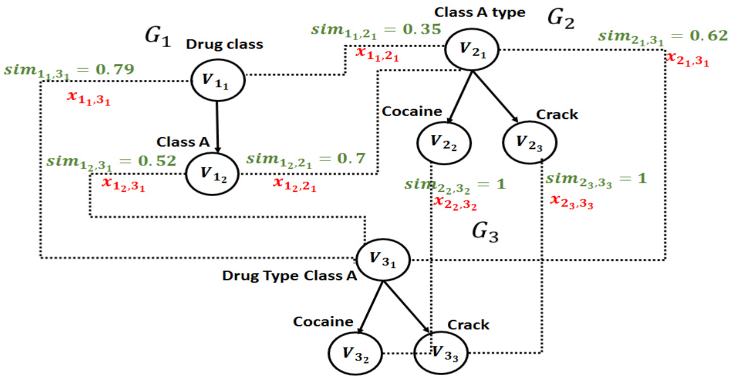


Fig. 1. A running example for holistic matching

Given the reductions between the schema matching and the graph matching problem, our contribution consists of using the linear programming technique to inject additional constraints on the graph matching problem to make it more suitable to meet the specificities of the schema matching problem. Our approach is divided into two steps:

- In the first step, we prepare LP4HM input data. According to the workflow described in the related work section, this step involves a pre-processing step for computing direction matrices, an element-level matching and aggregation phases for computing similarities between the elements of the different schemes;
- In the second step, we construct and execute LP4HM. According to the same workflow, this step ensures structural-level matching in the form of linear constraints as well as mapping extractions.

The following notations will be used in the remainder of this paper:

- N is the number of input graphs;

[1] http://www.scotland.gov.uk/Topics/Statistics/Browse/Crime-Justice/TrendData
http://data.gov.uk/dataset/seizures-drugs-england-wales

- i, j are the IDs of the graphs G_i and G_j;
- V_i is the set of vertices in the graph G_i;
- n_i is the cardinality of vertices ($|V_i|$) in the graph G_i;
- E_i is the set of edges in the graph G_i;
- k, l is the vertex order;
- v_{i_k} is the vertex of order k in the graph G_i;
- $e_{i_{k,l}}$ is the edge having as source v_{i_k} and as target v_{i_l} in the graph G_i.

3.1 LP4HM Input Data

Our linear program takes as input the following data: (1) a set of $N \geq 2$ graphs $G_i = (V_i, E_i)$, $i \in [1,N]$, (2) N direction matrices representing the hierarchical relationships between the elements of the graphs, (3) $N(N-1)/2$ similarity matrices representing an aggregated result of different element-level matchers.

Direction matrices. We compute a set of N direction matrices Dir_i of size $n_i \times n_i$ defined for each graph G_i, $\forall i \in [1,N]$. Each matrix encodes edge directions. It is defined as follows:

$$Dir_i = \{dir_{i_{k,l}},\ \forall k \times l \in [1,n_i] \times [1,n_i]\}$$

$$dir_{i_{k,l}} = \begin{cases} 1 & \text{if } e_{i_{k,l}} \in E_i \\ -1 & \text{if } e_{i_{l,k}} \in E_i \\ 0 & \text{otherwise} \end{cases}$$

Similarity matrices. We compute $N(N-1)/2$ similarity matrices denoted $Sim_{i,j}$ of size $n_i \times n_j$.

$$Sim_{i,j} = \{sim_{i_k,j_l}, \forall k \in [1,n_i], \forall l \in [1,n_j], \forall i \in [1,N-1], j \in [i+1,N]\}$$

For each pairwise graph G_i and G_j, a similarity measure sim_{i_k,j_l} is computed between all combination of labels of vertices v_{i_k} and v_{j_l} belonging respectively to G_i and G_j. These labels are first tokenized and stemmed, second different types of element-level matchers are applied on stemmed tokens, finally we apply the maximum as an aggregation function between the element-level matchers resulting measures.

We have selected different element-level matchers according to their time performance and quality in the recent comparative study of [23]. The selected metrics are as follows: (1) from the category character-based metrics we have chosen Edit distance, Monge-Elkan, Jaro-Winkler, ISUB and 3-gram to compute similarity between tokens and we have applied the generalized Mongue-Elkan [14] method on these metrics to get the similarity between concepts, (2) from the category token-based, we have applied Jaccard, soft TF-IDF with Levenshtein and (3) from the language-based category we have chosen Lin [15] and WUP [24]. WUP was not evaluated in [23] but it was emphasized as a well elaborate metric in [10]. We have used different libraries implementing these metrics: OntoSim[2], SimMetric[3], SecondString[4] and WS4J[5].

[2] http://ontosim.gforge.inria.fr/

[3] http://sourceforge.net/projects/simmetrics/

[4] http://secondstring.sourceforge.net/

[5] https://code.google.com/p/ws4j/

In this step, we also compute the value of the predefined threshold. For pairwise matching, this threshold corresponds to the median of all the maximums of rows in the similarity matrix. For the holistic matching, this threshold is the median of all local-threshold of each pairwise graph.

We acknowledge that our choices for the default threshold and the aggregation function are simplistic compared to other works in the literature, which focus more deeply on the problematic of matchers tuning and combination [10]. We point out that the main contribution of this paper resides in the holistic linear program of the following section.

3.2 Description of LP4HM

In this section, we describe the formalization of our linear program for holistic matching. The formalization is generalizable for $N \geq 2$ graphs. We will use the example of Fig.1 to illustrate some resulting constraints of LP4HM. We emphasize that to solve LP4HM for this example, the complete model is composed of the decision variables and the constraints of the three combinations (G_1, G_2), (G_1, G_3) and (G_2, G_3). Due to space consideration, we illustrate only the decision variables and the constraints of the combination (G_1, G_2) .

Decision Variable. We define a single decision variable which expresses the possibility to have or not a matching between two vertices belonging to two different input graphs. For each G_i and G_j, $\forall i \in [1, N-1]$, $j \in [i+1, N]$, x_{i_k, j_l} is a binary decision variable equals to 1 if the vertex v_{i_k} in the graph G_i matches with the vertex v_{j_l} in the graph G_j and 0 otherwise.

Example 1. In Fig.1, we have 6 decision variables between G_1 and G_2, the shown one are $x_{1_1,2_1}, x_{1_2,2_1}$, the not shown one are $x_{1_1,2_2}, x_{1_1,2_3}, x_{1_2,2_2}, x_{1_2,2_3}$.

Linear Constraints. Our linear program involves four constraints: C1 expresses the matching cardinality, we focus on 1:1 mapping cardinality [19], C2 constraints the selected mappings to be superior than a given threshold, C3 expresses sub/super class relations and C4 expresses coherence between edge directions.

C1 (Matching Cardinality) Resolving 1:1 mapping cardinality is equivalent to resolve a set of disjoint edges in the MWGM problem. To achieve this requirement, each vertex v_{i_k} in the graph G_i could match with at most one vertex v_{j_l} in the graph G_j, $\forall i \times j \in [1, N-1] \times [i+1, N]$. The corresponding constraint is as follows:

$$\sum_{l=1}^{n_j} x_{i_k, j_l} \leq 1, \quad \forall k \in [1, n_i]$$

Example 2. Applying C1 between G_1 and G_2 generates the following constraints:
$x_{1_1,2_1} + x_{1_1,2_2} + x_{1_1,2_3} \leq 1$
$x_{1_2,2_1} + x_{1_2,2_2} + x_{1_2,2_3} \leq 1$
$x_{1_1,2_1} + x_{1_2,2_1} \leq 1$
$x_{1_1,2_2} + x_{1_2,2_2} \leq 1$
$x_{1_1,2_3} + x_{1_2,2_3} \leq 1$

C2 (Matching Threshold) Usually matching systems [16][2] have to use a threshold in the mapping selection phase in order to enhance their matching quality. Our model handles this practice for a predefined threshold *thresh* computed as input data. The following constraint restricts the set of mapping solutions to be greater than a given *thresh*. $\forall i \times j \in [1, N-1] \times [i+1, N]$ and $\forall k \times l \in [1, n_i] \times [1, n_j]$

$$sim_{i_k,j_l}\, x_{i_k,j_l} \geq thresh\, x_{i_k,j_l}$$

Example 3. Applying C2 for a $thresh = 0.4$ between G_1 and G_2 generates the following constraints:
$0.35x_{1_1,2_1} \geq 0.4x_{1_1,2_1}$
$0.7x_{1_2,2_1} \geq 0.4x_{1_2,2_1}$

The decision variable $x_{1_2,2_1}$ can be affected to 0 or 1 as its similarity is higher than the value of the thresh. For a threshold equals to zero, our model is functional without this constraint. This is very interesting namely for high heterogeneous datasets.

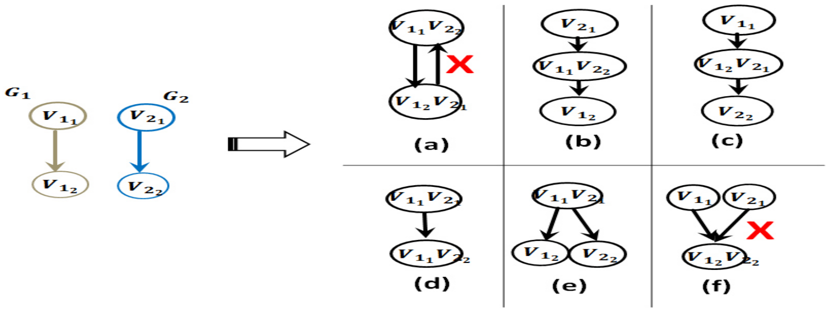


Fig. 2. An example of structural inconsistency

The second part of our constraints is related to structural matching. The idea is quite simple, it consists of prohibiting some cases of inconsistency by means of linear constraints. Fig.2 illustrates these cases, in the right side we have different possible matching solutions for the graphs given in the left side. The cases (a) and (f) represent incoherence in structural matching. Indeed, case (a) can be generated when parents match with children and vice versa. It gives rise to incoherence in edges directions. Case (f) can be generated when children match and their parents do not match. This case is known in schema matching literature [10] as super/sub class matching. It is particularity interesting for hierarchical structures in graph-like schemes. In the following, we propose the constraint C3 and C4 to resolve respectively the cases (f) and (a).

C3 (Super/sub class matching) This constraint makes parents match when their children match. For each pair of vertex v_{i_k} and v_{j_l} $\forall i \times j \in [1, N-1] \times [i+1, N]$ and $\forall k \times l \in [1, n_i] \times [1, n_j]$, their predecessors $v_{i_{pred(k)}}$ and $v_{j_{pred(l)}}$ have to match.

$$x_{i_k,j_l} \leq x_{i_{pred(k)},j_{pred(l)}}$$

Example 4. Applying C3 between G_1 and G_2 generates the following constraints:
$x_{1_2,2_2} \leq x_{1_1,2_1}$
$x_{1_2,2_3} \leq x_{1_1,2_1}$

C4 (Edges direction coherence) The purpose of this constraint is to prevent the generation of conflictual edges. By using direction matrices, the product of directions of the pairwise vertices has to be equal to 1 (i.e both edges have values -1 or 1). The constraint is expresses as follows: $\forall i \times j \in [1,N-1] \times [i+1,N]$ such as $i \neq j$ and $\forall k,k' \in [1,n_i]$ $\forall l,l' \in [1,n_j]$

$$x_{i_k,j_l} + x_{i_{k'},j_{l'}} + (dir_{i_{k,k'}} dir_{j_{l,l'}}) \leq 0$$

Example 5. Applying C4 between G_1 and G_2 generates the following constraints:
$x_{1_1,2_2} + x_{1_2,2_1} + dir_{1_1,1_2} dir_{2_2,2_1} \leq 0$
$x_{1_1,2_3} + x_{1_2,2_1} + dir_{1_1,1_2} dir_{2_3,2_1} \leq 0$
By substituting the direction values $dir_{1_1,1_2} = 1$, $dir_{2_2,2_1} = -1$, $dir_{2_3,2_1} = -1$ these constraints are equivalent to :
$x_{1_1,2_2} + x_{1_2,2_1} \leq 1$
$x_{1_1,2_3} + x_{1_2,2_1} \leq 1$
These constraints do not allow the matching of (v_{1_1}, v_{2_2}) and (v_{1_2}, v_{2_1}) in the same time (if it is the case we have $1+1 \leq 1$ which is impossible). An important aspect in this constraint is that it allows the feasibility of one of these matchings.

LP4HM model.

$$\begin{cases}
\max \displaystyle\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \sum_{k=1}^{n_i} \sum_{l=1}^{n_j} sim_{i_k,j_l}\, x_{i_k,j_l} \\
\\
s.t. \quad \displaystyle\sum_{l=1}^{n_j} x_{i_k,j_l} \leq 1,\ \forall k \in [1,n_i] \qquad\qquad (C1) \\
\qquad\qquad\qquad \forall i \in [1,N-1],\ j \in [i+1,N] \\
\\
\qquad sim_{i_k,j_l}\, x_{i_k,j_l} \geq thresh\, x_{i_k,j_l} \qquad\qquad (C2) \\
\qquad\qquad\qquad \forall i \in [1,N-1],\ j \in [i+1,N] \\
\qquad\qquad\qquad \forall k \in [1,n_i],\ \forall l \in [1,n_j] \\
\\
\qquad x_{i_k,j_l} \leq x_{i_{pred(k)},j_{pred(l)}} \qquad\qquad (C3) \\
\qquad\qquad\qquad \forall i \in [1,N-1],\ j \in [i+1,N] \\
\qquad\qquad\qquad \forall k \in [1,n_i],\ \forall l \in [1,n_j] \\
\\
\qquad x_{i_k,j_l} + x_{i_{k'},j_{l'}} - (dir_{i_{k,k'}} dir_{j_{l,l'}}) \leq 1 \qquad (C4) \\
\qquad\qquad\qquad \forall i \in [1,N-1],\ j \in [i+1,N] \\
\qquad\qquad\qquad \forall k,k' \in [1,n_i],\ \forall l,l' \in [1,n_j] \\
\\
\qquad x_{i_k,j_l} \in \{0,1\}\ \forall i \in [1,N-1],\ j \in [i+1,N] \\
\qquad\qquad\qquad \forall k \in [1,n_i],\ \forall l \in [1,n_j]
\end{cases}$$

A relaxation of LP4HM. The LP4HM program focuses on 1:1 mapping cardinalities using binary decision variables. We propose to relax the decision variables in the [0,1] interval. This relaxation enables resolving n:m matching cardinalities. Suppose that we have two vertices ”first name” and ”last name” both having the same similarity distance to ”name”. Therefore, we have two binary decision variables with the same similarity value, only one of these decision variables will be chosen. By relaxing variables in the [0,1] interval both variables will be assigned with a 0.5 value. We name LP4HM(relax) a relaxed version of LP4HM with decision variables in [0,1] resolving n:m mapping cardinalities.

4 Experimental Assessments

As far as we know, all existing benchmarks are devoted to pairwise schema matching problem. So, we have experimented our approach on a recent pairwise schema matching benchmark proposed by [6]. This benchmark is composed of ten datasets having different criteria and each one is composed of two XSD schemes. In this benchmark three tools have been compared: the generic matcher COMA++ [2], Similarity Flooding (SF) [16] and YAM [7]. SF was experimented with a threshold equals to 1. The experimentations of COMA++ have been done on three strategies (AllContext, FilteredContext and NoContext) and the best results have been maintained. The results of YAM correspond to an average of 200 runs per dataset.

Our approach is evaluated on two scenarios denoted ”A” and ”B”. Scenario ”A” consists of evaluating LP4HM and LP4HM(Relax) without the threshold constraint i.e we do not use the constraint C2. For this scenario, we denote our approaches as LP4HM_A and LP4HM_A(Relax). Scenario ”B” consists of evaluating LP4HM and LP4HM(Relax) with all the constraints. We have used a default threshold fixed to the median of the maximum similarity of each similarity matrix. This threshold is computed in the pre-processing step and inputted to the model. For this scenario, we denote our approaches as LP4HM_B and LP4HM_B(Relax). Our approach is resolved with an academic version of the CPLEX solver.

In this benchmark, the matching quality is evaluated according to precision, recall, f-measure, overall [16] and HSR (Human Spared Resources) [6]. Precision, recall and f-measure are well known measures issued from the information retrieval field. The overall measure is proposed by [16] and the HSR measure is proposed by [6]. Both measures compute post-match effort gained by the use of the matching tool.

4.1 Average Results

Table 1 and Fig.3 present average results for precision, recall, f-measure, overall and HSR of LP4HM and LP4HM(Relax) on strategies ”A” and ”B”, COMA++, SF and YAM. Without using a threshold in strategy ”A”, our approach has low precision results and high recall results particularly for the relaxed version. By using a threshold in strategy ”B”, the results of precision and recall of the relaxed and non relaxed versions, are more balanced which leads to balanced f-measure results. We can also observe that overall results are correlated to precision results. Indeed, if the precision is lower than or

Table 1. Average results of LP4HM, COMA++, SF and YAM

	Precision (%)	Recall (%)	F-Measure (%)	Accuracy (%)	HSR (%)
LP4HM_A	39	50	42	14	53
LP4HM_A(relax)	24	68	34	0	59
LP4HM_B	59	50	52	27	52
LP4HM_B(relax)	50	50	47	25	53
COMA++	66	36	43	28	34
SF	50	41	44	15	39
YAM	61	55	56	27	47

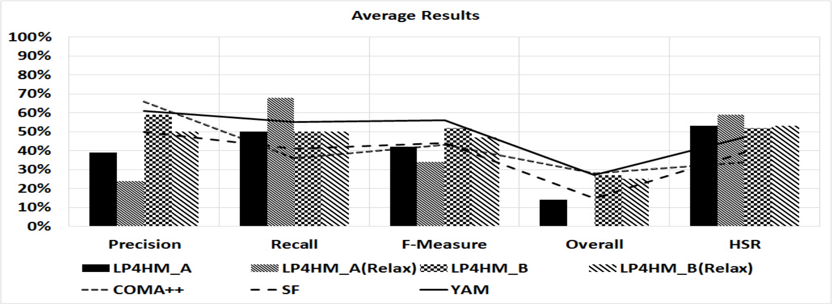


Fig. 3. A representation of the average results of LP4HM, COMA++, SF and YAM

equals to 50% so the overall results are lower than zero as reported in [16] (we note that all negative results have been rounded to 0 for all the approaches). Moreover, we notice that the HSR results are correlated to recall results as reported in [6]. The results of LP4HM_A(Relax) show more clearly these observations: the average precision is equal to 24%, the average recall is equal to 68% , the average overall is equal to 0% and the average HSR is equal to 59%. Compared to other approaches, LP4HM_A(Relax) has the worst precision and overall but the best recall and HSR. Otherwise, we notice that the results of LP4HM_B are close to the results of YAM, which is the best compared to COMA++ and SF.

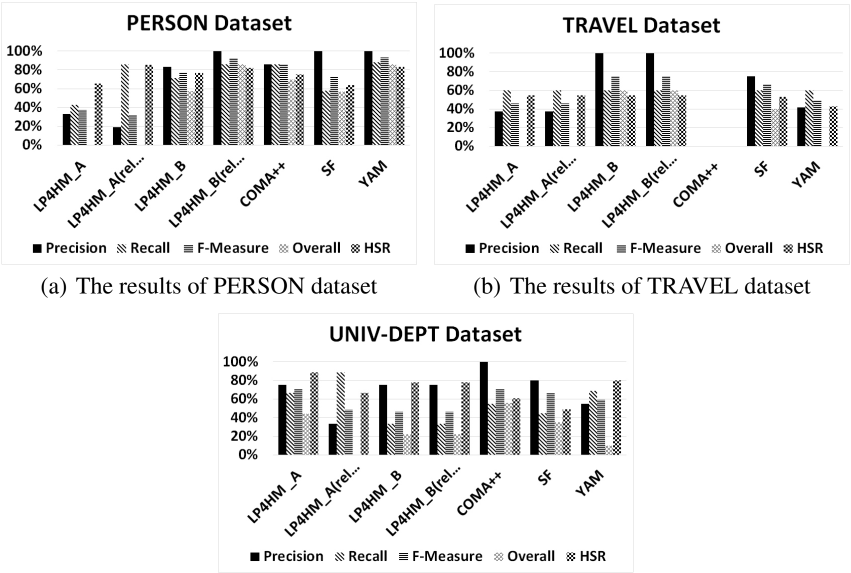
The comparison of the different strategies of our approach shows that the non relaxed versions LP4HM_A and LP4HM_B give a better compromise for all the quality measures. Using a pre-defined threshold for LP4HM_B enhances precision results of LP4HM_A . The recall still the same so we have better results on F-Measure and overall. Even if using a threshold improves the results of the different quality measure, we highlight that without using a threshold nor learning the HSR results of both LP4HM_A and LP4HM_A(Relax) are very interesting. These crucial results would be very interesting for holistic scenarios. The matching of $N \geq 2$ schemes is more difficult than matching 2 schemes. Indeed, if precision is better than recall so users have to find the missing mappings for N schemes simultaneously, which is a human difficult task. So when system returns good recall and low precision, users have just to eliminate the not relevant mappings proposed by the matcher.

In the next section, we will detail the results obtained for the different datasets.

4.2 Detailed Results

The authors of [6] have classified the different datasets according to five properties: label heterogeneity, domain specific, average size, structure and number of schemes. In this section, we will present the detailed results by grouping the datasets according to their average size. The average size represents the average number of schema elements. We will briefly report the type of the different properties for each dataset. Please refer to [6] for more details about the descriptions of these datasets.

Small Size Datasets (< 10 elements) The datasets PERSON, TRAVEL and UNIVERSITY DEPARTMENT (UNIV-DEPT) are small size datasets. PERSON dataset has low label heterogeneity and a nested structure. TRAVEL dataset has average label heterogeneity and a flat structure. UNIV-DEPT has high label heterogeneity and a flat structure.



(a) The results of PERSON dataset

(b) The results of TRAVEL dataset

(c) The results of UNIV-DEPT dataset

Fig. 4. Results of small size datasets.

The results of PERSON dataset are depicted in Fig.4(a). We notice that LP4HM_B (Relax) performs, without learning, the same results as YAM. LP4HM_B (Relax) and LP4HM_A (Relax) get the same best results of recall, but LP4HM_B(Relax) outperforms LP4HM_A(Relax) in precision. The recall of relaxed versions is better than the recall of not relaxed versions which is explained by the presence of 1:n expert mappings.

The results of TRAVEL dataset are depicted in Fig.4(b). We observe that the strategy ”B” of our version gives better results than strategy ”A”. But, it is worth to note that the results of recall are the same for the four versions. This implies that with only the proposed constraints our method is able to find 60% of expert mappings. For this dataset our approach is better than or equals to other approaches in recall and HSR.

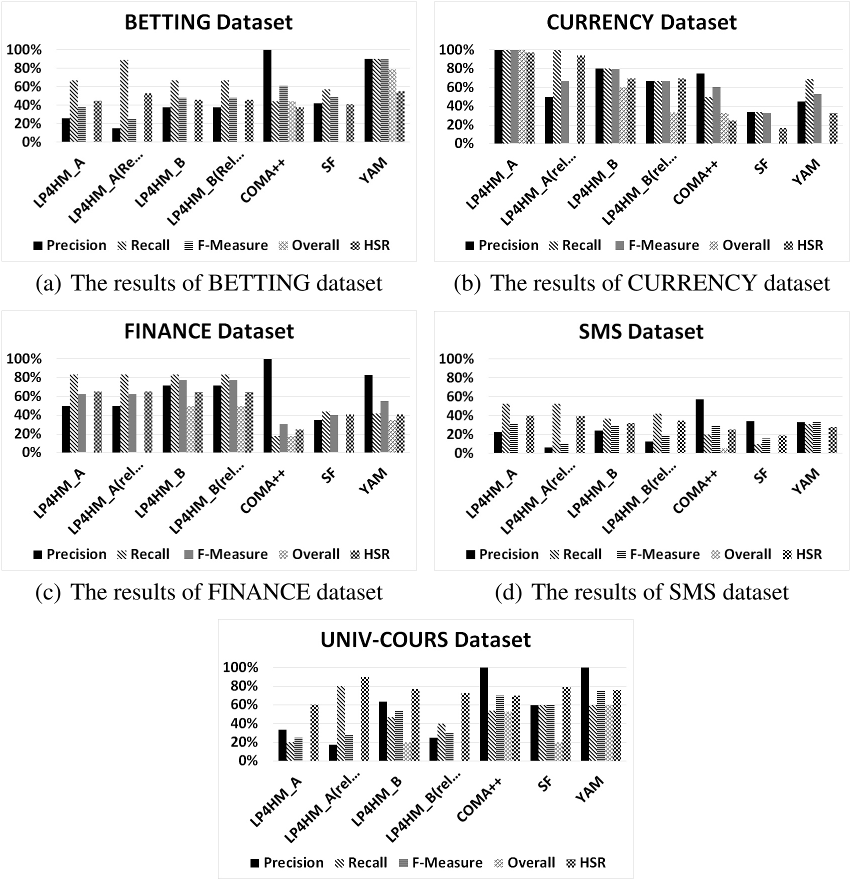
The results of UNIV-DEPT dataset are depicted in Fig.4(c). Contrarily to the two datasets described above, the strategy ”A” seems more efficient than strategy ”B”. Indeed, we can observe that recall results of strategy ”B” are worse than those in strategy ”A”. So experiments show that fixing a threshold in highly heterogeneous datasets will remove relevant solutions that have low similarity values. In this dataset, our approach is better than other approaches.

For small size datasets, our approach gives competitive results compared to COMA++, SF and YAM. For low and average heterogeneity, nested or flat structures, the two versions of our approach with a prefixed threshold are more effective. For high heterogeneity, the two versions of our approach without using a threshold nor learning on datasets are more effective.

Average Size Datasets (10 - 100 elements) The datasets BETTING, CURRENCY, FINANCE, SMS and UNIVERSITY COURSES (UNIV-COURS) are average size datasets. BETTING dataset has average label heterogeneity and a flat structure. CURRENCY dataset has average label heterogeneity and a nested structure. FINANCE is a domain specific dataset. It has average label heterogeneity and a flat structure. SMS dataset has average label heterogeneity and a flat structure.

The results of BETTING dataset are depicted in Fig.5(a). For this dataset, our results on recall and HSR are roughly the same as YAM. The prefixed threshold for strategy ”B” enhances precision but the recall still the same as the not relaxed version of the strategy ”A”. We can notice that LP4HM_A(Relax) performs 89% of recall and the worst precision. Contrarily to COMA++ whose precision attempts 100% but the recall is lower than 50%. The results of CURRENCY dataset are depicted in Fig.5(b). The results of our approach in the different versions outperform the results of COMA++, SF and YAM. Strategy ”A” of our approach is more effective than strategy ”B”, which demonstrates that using a threshold eliminates some relevant solutions. The results of FINANCE dataset are depicted in Fig. 5(c). Our approach outperforms the other approaches for all measures except precision measure. Recall is the same for all the versions of our approach. Strategy ”B” is little better than strategy ”A”. We remind that this dataset is a domain specific dataset, using Wordnet[6] as a dictionary is a good external resources. The results of SMS dataset are detailed in Fig.5(d). For this nested dataset strategy ”A” gives more important results on recall and HSR compared to strategy ”B”. Our approach is better than COMA++, SF and YAM for all measures except precision measure. The results of UNIV-COURS dataset are shown in Fig.5(e). Users gain at least 60% and at most 90% of human spared resources with not relaxed and relaxed versions of strategy ”A” of our approach. LP4HM_A(Relax) seems the more efficient strategy for this datasets, because 80% of relevant mappings have been detected and users have only to remove extra not relevant mappings.

[6] https://wordnet.princeton.edu/



(a) The results of BETTING dataset

(b) The results of CURRENCY dataset

(c) The results of FINANCE dataset

(d) The results of SMS dataset

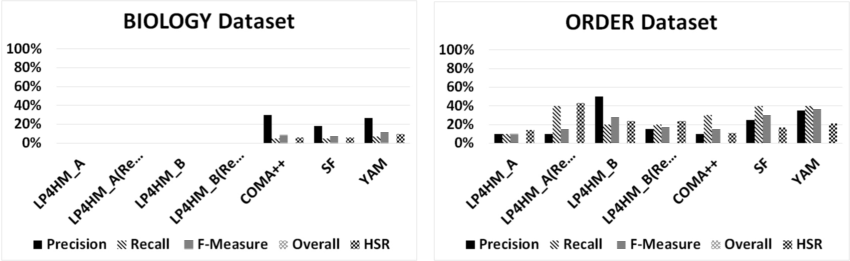
(e) The results of UNIV-COURS dataset

Fig. 5. The results of average size datasets

We can notice that all average size datasets have also average label heterogeneity. We have observed that for the nested structure datasets the strategy ”A” of our approach outperforms the strategy ”B”. This shows the efficiency of the structural constraints of our approach. For flat structure, we think that LP4HM_A(Relaxed) seems the best in recall and HSR. But LP4HM_B is more balanced.

Large Size Datasets (> 100 elements) The two large size datasets of this benchmark are BIOLOGY and ORDER. Both datasets have a nested structure. BIOLOGY is a domain specific dataset and has average label heterogeneity, while ORDER has low label heterogeneity.

Fig.6 depicts the results of BIOLOGY and ORDER datasets. In Fig.6(a), we notice that our approach fails to get relevant matchings for BIOLOGY dataset. We can also observe that the results of the other approaches are very low. As BIOLOGY dataset is



(a) The results of BIOLOGY dataset (b) The results of ORDER dataset

Fig. 6. The results of large size datasets

domain specific, our approach needs some external sources like specific dictionaries or a list of predefined synonyms. In Fig.6(b), we can observe that our approach especially LP4HM_A(Relax) successes to find 40% of the relevant mappings for ORDER dataset. Our results are similar to the results of SF and YAM.

The two large datasets, proposed in this benchmark, show the difficulties to maintain good quality results for large size datasets especially when the dataset is domain specific. We have also noticed that in these datasets an important number of sub-trees is repeated (due to references in XSD schemes). This is another difficulty in these datasets compared to the other datasets.

5 Conclusion

In this paper, we have presented a new approach to resolve the holistic matching of $N \geq 2$ schemes. Our approach is based on the linear programming technique to model the matching problem and its characteristics, such as the mapping cardinality and threshold filtering. Our model also holds constraints for the coherence of structural matching applied on hierarchical structures. We have experimented our approach on a recent benchmark in the literature [6]. The results of our approach are interesting for small and medium size datasets of different types of heterogeneity. We highlight that our approach returns good recall and HSR results without learning techniques nor threshold tuning. Our results are competitive compared to the results of existing approaches. Our future work will be devoted, first to experiment the quality of our model for other large size datasets with different types of aggregation functions and second to extend our model with further constraints to match labelled graphs.

References

1. Agreste, S., Meo, P.D., Ferrara, E., Ursino, D.: XML matchers: Approaches and challenges. Knowl.-Based Syst. 66, 190–209 (2014)
2. Aumueller, D., Do, H.H., Massmann, S., Rahm, E.: Schema and ontology matching with coma++. In: SIGMOD '05. pp. 906–908 (2005)

3. Berro, A., Megdiche, I., Teste, O.: A content-driven ETL processes for open data. In: New Trends in Database and Information Systems II - Selected papers of the 18th East European Conference on Advances in Databases and Information Systems and Associated Satellite Events, ADBIS 2014. pp. 29–40 (2014)
4. Berro, A., Megdiche, I., Teste, O.: Holistic statistical open data integration based on integer linear programming. In: RCIS' 2015. pp. 524–535 (2015)
5. Do, H.H., Rahm, E.: Matching large schemas: Approaches and evaluation. Information Systems 32(6), 857 – 885 (2007)
6. Duchateau, F., Bellahsene, Z.: Designing a benchmark for the assessment of schema matching tools. Open Journal of Databases (OJDB) 1(1), 3–25 (2014)
7. Duchateau, F., Coletta, R., Miller, R.J.: Yam: a schema matcher factory. In: CIKM. pp. 2079–2080 (2009)
8. Edmonds, J.: Maximum matching and a polyhedron with 0, 1-vertices. Journal of Research of the National Bureau of Standards B 69, 125–130 (1965)
9. Euzenat, J., Shvaiko, P.: Ontology matching. Springer-Verlag, Heidelberg (DE) (2007)
10. Euzenat, J., Shvaiko, P.: Ontology matching. Springer-Verlag, Heidelberg (DE), 2nd edn. (2013)
11. Euzenat, J., Valtchev, P.: Similarity-based ontology alignment in owl-lite. In: Proc. 16th european conference on artificial intelligence (ECAI). pp. 333–337. IOS press (2004)
12. Giunchiglia, F., Yatskevich, M., Shvaiko, P.: Semantic matching: Algorithms and implementation. JOURNAL ON DATA SEMANTICS 1, 2007 (2007)
13. Huber, J., Sztyler, T., Nner, J., Meilicke, C.: Codi: Combinatorial optimization for data integration: results for oaei 2011. In: OM. CEUR Workshop Proceedings, vol. 814. CEUR-WS.org (2011)
14. Jimenez, S., Becerra, C., Gelbukh, A., Gonzalez, F.: Generalized mongue-elkan method for approximate text string comparison. In: Gelbukh, A. (ed.) Computational Linguistics and Intelligent Text Processing, Lecture Notes in Computer Science, vol. 5449, pp. 559–570. Springer Berlin Heidelberg (2009)
15. Lin, D.: An information-theoretic definition of similarity. In: In Proceedings of the 15th International Conference on Machine Learning. pp. 296–304. Morgan Kaufmann (1998)
16. Melnik, S., Garcia-Molina, H., Rahm, E.: Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In: Proceedings of the 18th International Conference on Data Engineering. pp. 117–. ICDE '02, IEEE Computer Society (2002)
17. Niepert, M., Meilicke, C., Stuckenschmidt, H.: A Probabilistic-Logical Framework for Ontology Matching. In: Proceedings of the 24th AAAI Conference on Artificial Intelligence. pp. 1413–1418. AAAI Press (2010)
18. Rahm, E.: Towards large-scale schema and ontology matching. In: Bellahsene, Z., Bonifati, A., Rahm, E. (eds.) Schema Matching and Mapping, pp. 3–27. Data-Centric Systems and Applications, Springer Berlin Heidelberg (2011)
19. Rahm, E., Bernstein, P.A.: A survey of approaches to automatic schema matching. VLDB JOURNAL 10 (2001)
20. Schrijver, A.: Combinatorial Optimization - Polyhedra and Efficiency. Springer (2003)
21. Shvaiko, P., Euzenat, J.: Ontology matching: State of the art and future challenges. IEEE Trans. Knowl. Data Eng. 25(1), 158–176 (2013)
22. Shvaiko, P., Euzenat, J.: A survey of schema-based matching approaches. Journal on Data Semantics 4, 146–171 (2005)
23. Sun, Y., Ma, L., Shuang, W.: A comparative evaluation of string similarity metrics for ontology alignement. Journal of Information & Computational Science 12(3), 957 – 964 (2015)
24. Wu, Z., Palmer., M.: Verb semantics and lexical selection. In: In 32nd. Annual Meeting of the Association for Computational Linguistics, New Mexico State University, Las Cruces, New Mexico. pp. 133–138 (1994)