Ontology Matching

OM-2012

Proceedings of the ISWC Workshop

Introduction

Ontology matching[1] is a key interoperability enabler for the semantic web, as well as a useful tactic in some classical data integration tasks dealing with the semantic heterogeneity problem. It takes the ontologies as input and determines as output an alignment, that is, a set of correspondences between the semantically related entities of those ontologies. These correspondences can be used for various tasks, such as ontology merging, data translation, query answering or navigation on the web of data. Thus, matching ontologies enables the knowledge and data expressed in the matched ontologies to interoperate.

The workshop has three goals:

- To bring together leaders from *academia*, *industry* and *user institutions* to assess how academic advances are addressing real-world requirements. The workshop will strive to improve academic awareness of industrial and final user needs, and therefore direct research towards those needs. Simultaneously, the workshop will serve to inform industry and user representatives about existing research efforts that may meet their requirements. The workshop will also investigate how the ontology matching technology is going to evolve.
- To conduct an extensive and rigorous evaluation of ontology matching approaches through the OAEI (Ontology Alignment Evaluation Initiative) 2012 campaign[2]. The particular focus of this year's OAEI campaign is on real-world specific matching tasks involving, e.g., linked open data and biomedical ontologies. Therefore, the ontology matching evaluation initiative itself will provide a solid ground for discussion of how well the current approaches are meeting business needs.
- To examine similarities and differences from database schema matching, which has received decades of attention but is just beginning to transition to mainstream tools.

The program committee selected 6 submissions for oral presentation and 10 submissions for poster presentation. 21 matching system participated in this year's OAEI campaign.

[1] http://www.ontologymatching.org/
[2] http://oaei.ontologymatching.org/2012

The workshop included a panel entitled *What's the user to do? Ontology matching and the real world*, which addressed the following topics:

- What are the tools that are available today? What is their relationships with the mapping algorithms that the researchers are developing?
- What kind of support should ontology mapping user-facing tools provide?
- What is the quality of mappings that domain experts produce?
- Should domain experts be the ones performing ontology matching or are they so bad at it that all the mapping should be automatic anyway?
- How do we factor in the interactive tools into the formal evaluation such as OAEI?
- Are the ontologies mapped in OAEI representative of the ontologies that need to be mapping in the real world, or representative of the ontologies that the tool builders like to map?

with the following panelists:

- Kavitha Srinivas, IBM, USA;
- David Karger, MIT, USA;
- Jacco van Ossenbruggen, VU University Amsterdam, Netherlands;
- Jessica Peterson, Elsevier, USA.

Further information about the Ontology Matching workshop can be found at: `http://om2012.ontologymatching.org/`.

Acknowledgments. We thank all members of the program committee, authors and local organizers for their efforts. We appreciate support from the Trentino as a Lab (TasLab)[3] initiative of the European Network of the Living Labs[4] at Informatica Trentina SpA[5], the EU SEALS (Semantic Evaluation at Large Scale)[6] project and the Semantic Valley[7] initiative.

Pavel Shvaiko
Jérôme Euzenat
Anastasios Kementsietsidis
Ming Mao
Natasha Noy
Heiner Stuckenschmidt

November 2012

[3] http://www.taslab.eu
[4] http://www.openlivinglabs.eu
[5] http://www.infotn.it
[6] http://www.seals-project.eu
[7] http://www.semanticvalley.org/index_eng.htm

Organization

Organizing Committee

Pavel Shvaiko, TasLab, Informatica Trentina SpA, Italy
Jérôme Euzenat, INRIA & LIG, France
Anastasios Kementsietsidis, IBM Research, USA
Ming Mao, eBay, USA
Natasha Noy, Stanford University, USA
Heiner Stuckenschmidt, University of Mannheim, Germany

Program Committee

Michele Barbera, SpazioDati, Italy
Chris Bizer, Free University Berlin, Germany
Olivier Bodenreider, National Library of Medicine, USA
Marco Combetto, Informatica Trentina, Italy
Jérôme David, INRIA & LIG, France
Alfio Ferrara, University of Milan, Italy
Fausto Giunchiglia, University of Trento, Italy
Bin He, IBM, USA
Wei Hu, Nanjing University, China
Ryutaro Ichise, National Institute of Informatics, Japan
Antoine Isaac, Vrije Universiteit Amsterdam & Europeana, Netherlands
Krzysztof Janowicz, University of California, USA
Anja Jentzsch, Free University Berlin, Germany
Ernesto Jiménez-Ruiz, University of Oxford, UK
Yannis Kalfoglou, Ricoh Europe plc, UK
Patrick Lambrix, Linköpings Universitet, Sweden
Monika Lanzenberger, Vienna University of Technology, Austria
Rob Lemmens, ITC, The Netherlands
Maurizio Lenzerini, University of Rome La Sapienza, Italy
Vincenzo Maltese, University of Trento, Italy
Fiona McNeill, University of Edinburgh, UK
Christian Meilicke, University of Mannheim, Germany
Peter Mork, Noblis, USA
Nico Lavarini, Cogito - Expert System, Italy
Axel-Cyrille Ngonga Ngomo, University of Leipzig, Germany
Andriy Nikolov, Open University, UK
Leo Obrst, The MITRE Corporation, USA
Yefei Peng, Google, USA
François Scharffe, LIRMM, France
Luciano Serafini, Fondazione Bruno Kessler - IRST, Italy
Kavitha Srinivas, IBM, USA

Umberto Straccia, ISTI-C.N.R., Italy
Ondřej Šváb-Zamazal, Prague University of Economics, Czech Republic
Cássia Trojahn, INRIA & LIG, France
Raphaël Troncy, EURECOM, France
Giovanni Tummarello, Fondazione Bruno Kessler - IRST, Italy
Lorenzino Vaccari, European Commission - Joint Research Center, Italy
Ludger van Elst, DFKI, Germany
Shenghui Wang, Vrije Universiteit Amsterdam, Netherlands
Baoshi Yan, LinkedIn, USA
Songmao Zhang, Chinese Academy of Sciences, China

Table of Contents

PART 1 - Technical Papers

PART 2 - OAEI Papers

PART 3 - Posters

SLINT: A Schema-Independent Linked Data Interlinking System

Khai Nguyen[1], Ryutaro Ichise[2], and Bac Le[1]

[1] University of Science, Ho Chi Minh, Vietnam
{nhkhai,lhbac}@fit.hcmus.edu.vn
[2] National Institute of Informatics, Tokyo, Japan
ichise@nii.ac.jp

Abstract. Linked data interlinking is the discovery of all instances that represent the same real-world object and locate in different data sources. Since different data publishers frequently use different schemas for storing resources, we aim at developing a schema-independent interlinking system. Our system automatically selects important predicates and useful predicate alignments, which are used as the key for blocking and instance matching. The key distinction of our system is the use of weighted co-occurrence and adaptive filtering in blocking and instance matching. Experimental results show that the system highly improves the precision and recall over some recent ones. The performance of the system and the efficiency of main steps are also discussed.

Keywords: linked data, schema-independent, blocking, interlinking.

1 Introduction

Years of effort in building linked data has brought a huge amount of data in the LOD. However, maximizing the efficiency of linked data in the development of semantic web is still facing many difficulties. One of the current challenges is to integrate the individual data sources for building a common knowledge system. When different data source may contain heterogeneous instances, which co-refer to the same real-world objects, data integration process requires the detection of such objects to ensure the integrity and consistency of data. Detecting all identities between data sources is the mission of data interlinking. Data interlinking consist of two main steps, blocking and instance matching. While blocking aims at pruning the number of comparison, instance matching is to determine the matching status of two interested instances.

Current interlinking methods can be categorized into two main groups: schema-dependent [2,7,10] and schema-independent [1,3,4,9]. The former requires the knowledge about meaning of RDF predicates (e.g. predicate #preLabel declares the label of object) and the predicate alignments (e.g. predicate #preLabel matched with predicate #name). In contrast, the latter does not need these information, therefore it does not rely on human knowledge about the schema. Because a linked data instance is a set of many RDF triples (subject, predicate, object), the schema of a data source refers to the list of all used predicates, which are closely related to vocabulary and ontology. The schemas are usually different for each data sources, even in the same data source but different domains. Clearly, schema-independent methods are more applicable when it can

work on every kind of source or domain without any human's instruction. Besides, manual specifications of interlinking rules frequently ignore the hidden useful predicate alignments.

We present SLINT system, which use a new approach for schema-independent linked data interlinking. SLINT automatically selects important RDF predicates using the coverage and discriminability. The selected predicates are combined to construct the predicate alignments in conciliation of data type. We estimate the confidence of predicate alignments to collect the most appropriate alignments for blocking and interlinking. By this way, the collective information of instance is frequently leveraged. Blocking is therefore more complete, compact, and supportive for interlinking. Also, we apply adaptive filtering techniques for blocking and instance matching. In experiment, we compare SLINT with three systems, which participated OAEI 2011 instance matching campaign, and report the high improvement on both precision and recall. Experiments on the performance of SLINT and the efficiency of blocking step are also reported.

The paper is organized as follow: the next section is the overview of previous work. Section 3 describes the detail of SLINT system. Section 4 reports our experimental evaluation. Section 5 closes the paper with conclusion and outlook.

2 Related work

Data interlinking is an early studied area, however, this problem in linked data has just been recently attended. Silk [10], a well-known framework, provides a declarative interface for user to define the predicate alignments as well as the similarity metrics for matching. Silk was used as a main component of the LDIF [8], a multiple linked data sources integration framework. Recently, Isele and Bizer have improved their Silk by applying an automatic linkage rules generation using genetic algorithm [3]. The work is very interesting at the modeling of appropriate fitness function and specific transformations for genetic programming in the context of interlinking. This work makes Silk to be schema-independent. With the similar objective, RAVEN [4] minimize human curation effort using active learning, while Nikolov et al. also use genetic algorithm with the research target is an unsupervised learning process [6]. Also with schema-independent goal, Nguyen et al. suggest using decision tree classifier for determining the matching status of two instances [5].

Zhishi.Links [7] is one of the current state-of-the-art matchers. This system adopt the idea of Silk's pre-matching step, by using label of objects such as *skos:prefLabel* or *scheme:label*, to group similar instances. Afterward, a more complex semantic similarity is utilized for matching. This system ranks first at the OAEI 2011 instance matching, while the second best is SERIMI [1], a schema-independent system. SERIMI selects RDF predicates and predicate alignments using entropy and RDF object similarity, respectively. AgreementMaker [2] is an ontology matching and instance matching system. It firstly generates candidates set by comparing the labels of instances. These candidates are then divided into smaller subsets, in which every pair is matched to produce the final alignments.

Most of previous interlinking systems do not deeply investigate on blocking step, which generates potential identity pairs of instances. Song and Heffin focus

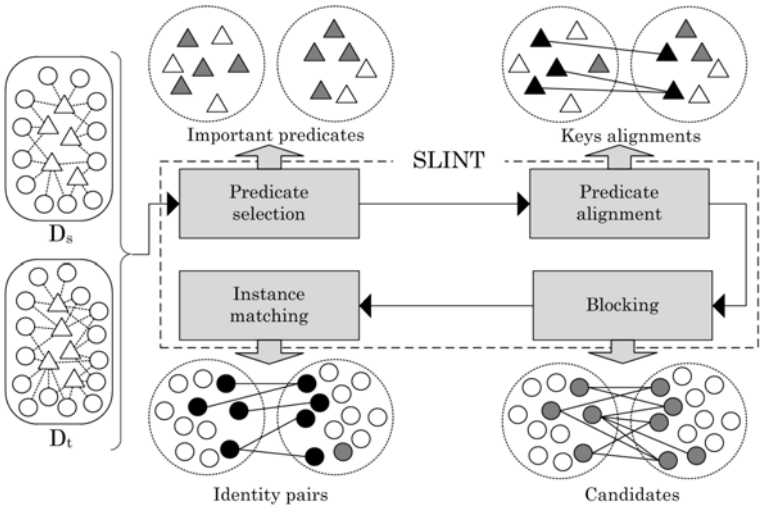


Fig. 1. Summary of data interlinking process

on blocking scheme for linked data interlinking for parallel independent work [9]. It is a very interesting idea when the authors propose an unsupervised learning for maximizing the usefulness of blocking keys, which are the combinations of RDF predicates. The authors conduct experiments on some large datasets, which also proof for the scalability.

In general, the schema-dependent approaches compare two instances by specified properties. That is, they can detect almost right identity pairs but the precision may be low on highly ambiguous data sources. The reason is that some useful information can be ignored since the manual predicate alignment frequently is not an optimal solution. In contrast, schema-independent approaches reconcile precision and recall because of elaborate analysis on the data. Although these approaches need to collect predicate alignments, the matching is more effective when collective information is frequently used. Comparing SLINT with previous interlinking systems, the prominent differences are the predicate selection, predicate alignment, and adaptive filtering for blocking and interlinking. In the next section, we describe these elements as the details of SLINT.

3 Schema-independent linked data interlinking system

This section describes the SLINT system. The overview of the interlinking process for source data D_s and target data D_t is shown in Figure 1. In this figure, the small circles and triangles respectively stand for instances and theirs RDF predicates. The referred circles of each step are the output of that step. The SLINT system consists of four steps. The interlinking process begins with *predicate selection*, which collects the *important predicates* from all predicates of each data sources. In the second step, *predicate alignment*, selected predicates are combined in accordance with their data type to construct the raw predicate alignments. We estimate the confidence of every raw alignment to measure its appropriateness. A raw alignment will be called a key alignment if its confidence

satisfies a filtering condition. These *key alignments* provide much useful information in blocking and instance matching steps. Next, *blocking* step is designed to reduce the number of comparison by producing identity *candidates* of instances. The *instance matching* afterward only need to verifies the retrieved candidates for discovering the *identity pairs*. The followings are the detail of each step.

3.1 Predicate selection

The mission of this step is to find the important predicates from the schema, which consists of all predicates appearing in interested data source. We use two criteria for determining the importance level of predicate p: coverage $cov(p, D)$ and discriminability $dis(p, D)$. Eq.1 and Eq. 2 are the explanations of these criteria when considering predicate p of data source D.

$$cov(p, D) = \frac{|\{x|\exists < s, p, o >\in x, x \in D\}|}{|D|}. \quad (1)$$

$$dis(p, D) = \frac{|\{o|\exists x \in D, < s, p, o >\in x\}|}{|\{< s, p, o > |\exists x \in D, < s, p, o >\in x\}|}. \quad (2)$$

In these equation, x represents an instance and is a set of RDF triple $< s, p, o >$ (subject, predicate, object). D is the interested data source and is a set of instances. From each input source, we collect the predicates having high score of coverage and discriminability. A predicate p is selected if it satisfies the condition in Eq.3, which inherits from the idea of [9].

$$(cov(p, D) \geq \alpha) \wedge (dis(p, D) \geq \beta) \wedge (HMean(cov(p, D), dis(p, D)) \geq \gamma). \quad (3)$$

The α and β imply the minimum standard of an important predicate, whereas γ, the condition for harmonic mean of $dis(p, D)$ and $cov(p, D)$, is the main requirement. Therefore, we set small values for α and β and larger value for γ.

Song and Heffin focus on learning blocking key by iteratively maximize the coverage and discriminability of the set of predicates [9]. In our system, we use the same discriminability function with theirs and slightly different function for coverage. For the numerator of Eq. 1, while they use the number of RDF subjects, we use the number of instances, because we aim at finding the frequency of predicate over instances, not over RDF subjects.

Important predicates are expected to be used for declaring the common properties and distinct information of objects. Since coverage and discriminability respectively express the former and latter, the combination of them is therefore appropriate for the objective of predicate selection. If a predicate has a high coverage but a low discriminability or otherwise, it will not be important. An example for this kind of predicate is *rdf:type*. This predicate is frequently used but it usually describes a limit range of various RDF objects when observing the instances in the same domain.

3.2 Predicate alignment

In this step, we find the appropriate alignments of predicates between the source data and target data. An alignment of two predicates is considered to be appropriate if the interested predicates describe the similar properties of instances.

From selected predicates of source data and target data, we connect every type-matched pair and select the alignments whose confidence is higher than threshold δ. Selected predicate alignments are called key alignments. The confidence of an alignment is the Dice coefficient between the representatives of RDF objects described by its formed predicates. Eq. 4 is the equation of confidence $conf(p_s, p_t)$ for the alignment between predicate p_s in source data D_s and predicate p_t in target data D_t.

$$conf(p_s, p_t) = \frac{2 \times |R(O_s) \cap R(O_t)|}{|R(O_s)| + |R(O_t)|}, O_k = \{o | \exists x \in D_k, < s, p_k, o > \in x\}. \quad (4)$$

In above equation, R is the function that returns the representative elements of RDF objects. The return values of R depend on the type of predicates. We divide the predicates into five different types: *string*, *URI*, *decimal*, *integer*, and *date*. This separation is based on the variety of data types in the real world and covers most of current types of linked data. For *string*, we extract the word token of RDF objects. For *URI*, we omit the domain part and use the same manner as for *string*, with the assumption that slash '/' is the token separator. For *decimal*, we take the 2-decimal digits rounded values. For *integer* and *date*, we do not transform the RDF objects and use the original values. For determining type of a predicate, we use the major type of RDF objects declared by this predicate. For example, if 51% appearance times of p is to describe *decimal* values, the data type of p will be *decimal*. Currently, we detect the type of RDF objects without the consideration about the difference in their metric (e.g. the units of time, distance, area).

The confidence of an alignment represents the similarity of RDF objects between two data sources. The predicates having the same meaning frequently describe the similar information. Therefore, alignments of matched predicates usually have higher confidence than the others. It means that a predicate satisfying the requirements of an important predicate is verified again, by considering the confidence of all alignments in which it appears. For example, the predicate *rdfs:comment* has possibility to be important but the confidences of its alignments are usually low because the denominator of Eq.3 is very high in this case.

A common limiting point of almost previous systems is the use of string measurement for every type of RDF objects. Clearly, this approach is not sufficient to cover the meaning of RDF objects, thus, does not well estimate the similarity of non-string values. We discriminate data types not only in combining predicates, but also in blocking and instance matching.

It is not easy for a person to detect all useful predicate alignments, this step is therefore very meaningful, and in accompaniment with predicate selection, it tackles the schema-independent goals. The next steps are the use of selected key alignments and their formed predicates.

3.3 Blocking

As we introduced, the blocking aims at retrieving the candidates for instance matching step by grouping similar instances into the same block. A candidate is a pair of two instances, one belongs to source data and one belongs to target data. The blocking can be divided into three sub phases. The first phase indexes

Algorithm 1: Generating candidates set

Input: D_s, D_t, Pr_s, Pr_t, ζ, ϵ
Output: Candidate set C

```
1  H ← ∅
2  M[|D_s|, |D_t|] ← {0}
3  C ← ∅
4  foreach < D, P >∈ {< D_s, Pr_s >, < D_t, Pr_t >} do
5      foreach x ∈ D do
6          foreach p_i ∈ P do
7              sumConf ← Σ_{p_j ∈ {Pr_s, Pr_t} \ P} conf(p_i, p_j)
8              foreach r ∈ Rp(O), O = {o| < s, p_i, o >∈ x} do
9                  if not H.ContainsKey(r.Label) then
10                     H.AddKey(r.Label)
11                 H.AddValue(r.Label, D, < x, r.Value × sumConf >)
12 foreach key ∈ H.AllKeys() do
13     foreach < x_s, v_s >∈ H.GetValues(key, D_s) do
14         foreach < x_t, v_t >∈ H.GetValues(key, D_t) do
15             M[x_s, x_t] ← M[x_s, x_t] + v_s × v_t
16 foreach x_s ∈ D_s do
17     foreach x_t ∈ D_t do
18         max_s ← Max(M[x_s, x_j]), ∀x_j ∈ D_t
19         max_t ← Max(M[x_i, x_t]), ∀x_i ∈ D_s
20         max ← HMean(max_s, max_t)
21         if M[x_s, x_t] ≥ ζ and M[x_s, x_t] / max ≥ ε then
22             C ← C ∪ < x_s, x_t >
23 return C
```

every instance in each data source by the value extracted from its RDF objects. The second phase traverses the index table and builds a weighted co-occurrence matrix. The final phase uses this matrix as the input information when it applies a filtering technique to select candidates. Algorithm 1 is the pseudo-code of the whole blocking process. In this algorithm, Pr_s and Pr_t represent the list of predicates that form the key alignments, where Pr_k belongs to D_k. H, M, C, Rp represent the inverted-index table, weighted co-occurrence matrix, candidates set, and representative extraction method, respectively.

The lines 4-11 perform the invert-indexing, a well-known indexing technique. By once traversing each data source, we extract the representatives of RDF objects and use them as the keys of invert-index table. An element r in the representatives set of RDF objects consists of two fields: the label $r.Label$ and value $r.Value$. While $r.Label$ is the return value of representative extraction

method R as in predicate alignment step, $r.Value$ is computed in accordance with the data type of predicate p_i. If p_i is *string* or *URI*, we set the value to TF-IDF score of the token. If p_i is either *decimal*, *integer*, or *date*, we assign the value to a fixed number, which is 1.

After constructing the invert-index table, we compute weighted co-occurrence matrix M as the lines 12-15, by accumulating the value for each matrix element.

The lines 16-22 are the process of adaptive filtering. An instance pair $< x_s, x_t >$ will be considered as a candidate if its weighted co-occurrence value $M[x_s, x_t]$ satisfies the threshold ϵ, after divided for the harmonic mean of maximum weighted co-occurrences of x_s and x_t. In addition, we use ζ, a small threshold, to avoid the surjection assumption. The identities frequently have the high weighted co-occurrences; however, these values are variable for different pairs. Choosing a fixed threshold for selecting candidates is not good in this situation and is a tedious task. Therefore, we use the coefficient of $M[x_s, x_t]$ and max, which is a data driven element and expresses the adaptive filtering idea.

Blocking is very important because it reduces the number of comparison in instance matching. However, it seems not to have been sufficiently attended when most of previous systems use quite simple method for blocking. In comparison with blocking step in previous interlinking systems, the key difference of our method is the weighted co-occurrence matrix and the adaptive filtering. While previous systems compare the pairs of RDF objects, we aggregate the product of the weight of their matched representatives. For candidate selection, Silk [10] and Zhishi.Links [7] use *top-k* strategy, which selects k candidates for each instance. The approach is very good for controlling the number of candidates, but determining the value of k is not easy. Song and Heffin use *thresholding* selection [9], which is also similar with SERIMI [1]. Our method also use thresholding approach as the availability of ζ. However, the key idea of our selection method is the adaptive filtering because the impact of ζ is not high. Frequently, there are many of non-identity pairs between two data sources, ζ is therefore usually configured with a low value.

Next, we input the set of candidates C and the key alignments A into the final step, instance matching.

3.4 Instance matching

The instance matching verifies the selected candidates to determine their identity state. We compute the matching *score* for every candidate and choose the ones that have high score as the identity pairs. For each element in A, we compute the similarity of RDF objects, which declared by the involved predicates of interested key alignment. The final score of two instances is the weighted average value of all these similarities, and the weights are the confidences of key alignments. Eq.5 is the computation of matching score between instance $x_s \in D_s$ and $x_t \in D_t$.

$$
\begin{aligned}
score(x_s, x_t) = & \frac{1}{W} \sum_{<p_s, p_t> \in A} conf(p_s, p_t) \times sim(R(O_s), R(O_t)), \\
Where \quad & O_k = \{o | \exists x \in D_k, < s, p_k, o > \in x\} \\
& W = \sum_{<p_s, p_t> \in A} conf(p_s, p_t).
\end{aligned} \tag{5}
$$

In this equation, R stands for the representative extraction methods, which are similar to those in predicate alignment step.

Categorizing five data types, we implement three different versions of *sim* function in accordance with the type of predicates. For *decimal* and *integer*, we take the variance of the values to remedy the slight difference of data representations. For *date*, the *sim* function yields 1 or 0 when the values are equal or not, respectively. A date is usually important if it is a property of the object (e.g. birthday, decease date, release date of a movie). Therefore, the exact matching is an appropriate selection for dates comparison. For *string* and *URI*, we compute the TF-IDF modified cosine similarity, as given in Eq.6. TF-IDF is used because its advantage in disambiguating the instances sharing common tokens. TF-IDF also minimizes the weight for the stop-words, which are usually useless.

$$sim(Q_s, Q_t) = \frac{\sum_{q \in Q_s \cap Q_t} TFIDF(q, Q_s) TFIDF(q, Q_t)}{\sqrt{\sum_{q \in Q_s} TFIDF^2(q, Q_s) \times \sum_{q \in Q_t} TFIDF^2(q, Q_t)}}. \quad (6)$$

Similar with blocking step, we do not use fixed single threshold for filtering the candidates. Two instances will be considered as an identity pair if their score is higher than the maximum score of the candidates in which either of instances appears. The final identities set I is formalized in Eq.7.

$$I = \{< x_s, x_t > | score(x_s, x_t) \geq \eta \wedge \frac{score(x_s, x_t)}{max_{\forall < x_m, x_n > \in C, x_m \equiv x_s \vee x_n \equiv x_t} score(x_m, x_n)} \geq \theta\}. \quad (7)$$

An identity pair is expected to be the highest correlated candidate of each instance. However, it usually is not true because the ambiguity of instances. A thresholding method that relies on the highest score would be better in this situation. While true identity pair and its impostors have the similar score, θ is assigned with a quite large value. On the other hand, η is additionally configured as the minimum requirement of an identity. Like ϵ in Algorithm 1, η ensures there is no assumption about the surjection of given data sources.

The key distinctions of our approach in comparison with the previous are the use of weighted average and adaptive filtering. Previous systems do not have the data driven information like confidence of key alignments. Silk [10] provides a manual weighting method; however, a good weight usually depends much on the human knowledge about the data. For identity selection, Zhishi.Links [7] and AgreementMaker[2] eventually select the best correlated candidates, while Silk [10] and SERIMI [1] use threshold-based selection. We compare the interlinking result of our system with those of Zhishi.Links, SERIMI, and AgreementMaker in our experiments, which are reported in the next section.

4 Experiment

4.1 Experiment setup

We evaluate the efficiency of blocking step and the whole interlinking process of SLINT. We also compare SLINT with Zhishi.Links [7], SERIMI [1], and

AgreementMaker [2], which recently participated OAEI 2011 instance matching campaign. Discussion on predicate selection and predicate alignment are also included. For every test in our experiment, we use the same value for each threshold. We set α, β ,γ (Eq. 3), δ (Eq. 4), ζ, ϵ (Algorithm 1), η, and θ (Eq. 7) to 0.25, 0.25, 0.5, 0.25, 0.1, 0.5, 0.25, and 0.95, respectively. The fixed values of α, β, γ and δ express the schema-independent capability of SLINT.

Like previous studies, for blocking, we use two evaluation metrics: pair completeness (PC) and reduction ratio (RR); For interlinking, we use recall (Rec), precision (Prec), and F1 score, the harmonic mean of recall and precision. Eq.8 and Eq.9, Eq.10, and Eq.11 are the computations of used metrics.

$$PC = \frac{Number\ of\ correct\ candidates}{Number\ of\ actual\ identity\ pairs}. \tag{8}$$

$$RR\ =\ 1\ -\ \frac{Number\ of\ candidates}{Number\ of\ all\ pairs}. \tag{9}$$

$$Rec\ =\ \frac{Number\ of\ correct\ identity\ pairs}{Number\ of\ actual\ identity\ pairs}. \tag{10}$$

$$Prec\ =\ \frac{Number\ of\ correct\ identity\ pairs}{Number\ of\ discovered\ pairs}. \tag{11}$$

The performance of an interlinking system is also very important. We report the execution times of SLINT when running on a desktop machine equipped with 2.66Ghz quad-core CPU and 4GB of memory.

4.2 Datasets and discussion on predicate selection & predicate alignment

We use 9 datasets in experiment. The first 7 datasets are IM@OAEI2011 datasets, the ones used in instance matching campaign at the OAEI 2011[3]. Concretely, we use the datasets of interlinking New York Times track, which asks participants to detect identity pairs from NYTimes to DBpedia, Freebase, and Geonames. These datasets belong to three domains: *locations*, *organizations*, and *people*. The IM@OAEI2011 datasets are quite small. Therefore, for evaluating the computational performance of the system, we select two larger datasets. The first one, a medium dataset, contains 13758 pairs in *film* domain between DBpedia and LinkedMDB[4]. The second one is a quite large dataset, which contains 86456 pairs in *locations* domain between DBpedia and Geonames[5]. All datasets are downloaded by dereferencing URI and stored in external memory in advance. We remove triples having *owl:sameAs* predicates and *rdf:seeAlso* predicates of course. Table 1 gives the overview of these datasets. In this table, IM@OAEI2011 datasets are from D1 to D7, and the last two datasets are D8 and D9. We also include the number of predicates and predicate alignments in this table. Denotes that s and t are the source and target data, respectively; Pr_d and Pf_d are the

[3] http://oaei.ontologymatching.org/2011/instance/
[4] http://downloads.dbpedia.org/3.7/links/linkedmdb_links.nt.bz2
[5] http://downloads.dbpedia.org/3.7/links/geonames_links.nt.bz2

Table 1. Number of predicates and predicate alignments

ID	Source	Target	Domain	Pairs	Pr_s	Pr_t	Pf_s	Pf_t	A	K
D1	NYTimes	DBpedia	Locations	1920	12	2859	6	24	26	7
D2	NYTimes	DBpedia	Organizations	1949	10	1735	6	14	21	5
D3	NYTimes	DBpedia	People	4977	10	1941	5	20	33	8
D4	NYTimes	Freebase	Locations	1920	12	775	6	18	20	4
D5	NYTimes	Freebase	Organizations	3044	10	1492	5	18	33	3
D6	NYTimes	Freebase	People	4979	10	1844	5	18	32	5
D7	NYTimes	Geonames	Locations	1789	12	32	6	13	14	4
D8	DBpedia	LinkdMDB	Movie	13758	1343	54	16	7	31	14
D9	DBpedia	Geonames	Locations	86456	8194	17	11	7	27	8

number of predicates in data source d before and after selected by predicate selection step, respectively; A and K are the number of all predicate alignments and only key alignments, respectively.

In general, excepts in NYTimes, the number of available predicates in the schema of each data source is very large, but the important predicates occupy a very small percent. As our observation, the predicates declaring the label or the name of objects are always aligned with a very high confidence. The non-string type predicates also frequently construct the key alignments. For example, in *locations* domain, the key alignments always contain the right combination of predicates declaring latitudes and longitudes. The predicate *releaseDate* of DBpedia is successfully combined with predicate *date* and predicate *initial_relase_date* of LinkedMDB. An interesting key alignment in dataset D6 is the high confidence combination of *core#preLabel* of NYTimes and *user.mikeshwe.default_domain.videosurf_card.videosurf_link_text* of Freebase. The latter predicate may be difficult for manual selection since the meaning of the predicate name does not imply the label. Clearly, it is not easy for human to detect every compatible predicates. When manually doing this task, we may lose to leverage all useful information.

4.3 Blocking and interlinking result

This section reports the result of blocking and the whole interlinking process. Concretely, we report the pair completeness and reduction ratio of blocking, and precision, recall, F1 score, and the runtime of the system. Table 2 shows these metrics on each dataset. According to this table, although we cannot retain all identity pairs, the lowest PC is still very high at 0.94. Besides, the high RRs reveal that the numbers of retrieved candidates are very small if compared with the numbers of total pairs. For all the evidences of PC and RR, the aim of blocking is successfully achieved.

For interlinking, the precison and recall are very competitive. The recall, which is not much lower than pair completeness, implies that the instance matching performs a good work. The high precision implies that our system has a efficient disambiguation capability on tested datasets. It seems easy for SLINT to interlink *people* domain, whereas in *locations* domain, SLINT achieves the best result on IM@OAEI2011 datasets involving with Geonames.

The execution time of SLINT is very good in overview. Because we use co-occurrence structure in blocking, the memory on tested machine cannot satisfy

Table 2. Number of candidates, PC, RR, Rec, Prec, F1, and execution time

Dataset	Blocking			Interlinking			
	Candidates	PC	RR	Prec	Rec	F1	Runtime
D1	4102	0.9901	0.9989	0.9636	0.9651	0.9644	3.55
D2	3457	0.9831	0.9970	0.9768	0.9487	0.9625	4.29
D3	9628	0.9950	0.9972	0.9883	0.9841	0.9862	12.74
D4	3580	0.9849	0.9990	0.9486	0.9521	0.9504	3.78
D5	7744	0.9823	0.9992	0.9610	0.9560	0.9585	6.71
D6	10333	0.9938	0.9996	0.9944	0.9904	0.9924	18.25
D7	2473	0.9961	0.9959	0.9883	0.9888	0.9885	1.63
D8	33926	0.9948	0.9998	0.9317	0.9868	0.9584	67.76
D9	418592	0.9468	0.9999	0.9782	0.9418	0.9596	2465.38

a very large dataset. In our context, interlinking dataset D9 has such issue. We temporarily implement a parallel program for re-computing every element of the co-occurrence matrix. The interlinking on this dataset is therefore takes much time because the repeat of data traversing and high computational cost. However, the high speeds on other datasets are really promising for scaling-up our system.

The advantage of blocking is very high if we compare the time of interlinking with and without this step. For example, the time for instance matching step to match 33926 candidates of dataset D8 is 12.2 seconds. It means that the time for matching all available pairs will be nearly 17 hours, whereas this number is only 67.76 seconds in total if we implement the blocking step. Blocking averagely occupies 58% total runtime of interlinking process on the nine tested datasets. Although this number is over a half, the advantage of blocking is still very considerable.

4.4 Comparison with previous interlinking systems

As mentioned, we compare our system with AgreementMaker [2], SERIMI [1], and Zhishi.Links [7]. Because these systems recently participated instance matching campaign of the OAEI 2011, we use the results on IM@OAEI2011 datasets for comparison. Table 3 shows the interlinking result of SLINT and others. As showed in this table, it is clear that our system totally outperforms the others on both precision and recall. AgreementMaker has a competitive precision with SLINT on dataset D3 but this system is much lower in recall. Zhishi.Links results on dataset D3 are very high, but the F1 score of SLINT is still 0.05 higher in overall.

The prominent differences of SLINT and these systems are that we use the confidence of alignment as the weight for blocking and instance matching, and discriminate data types with the use of TF-IDF for the token of string and URI. Generally, SLINT is verified as the best accurate one among compared systems.

5 Conclusion

In this paper, we present SLINT, an efficient schema-independent linked data interlinking system. We select important predicates by predicate's coverage and discriminability. The predicate alignments are constructed and filtered for obtaining key alignments. We implement an adaptive filtering technique to produce

Table 3. Comparison with previous interlinking systems.

Dataset	SLINT			Agree.Maker			SERIMI			Zhishi.Links		
	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
D1	**0.96**	**0.97**	**0.96**	0.79	0.61	0.69	0.69	0.67	0.68	0.92	0.91	0.92
D2	**0.98**	**0.95**	**0.96**	0.84	0.67	0.74	0.89	0.87	0.88	0.90	0.93	0.91
D3	**0.99**	**0.98**	**0.99**	0.98	0.80	0.88	0.94	0.94	0.94	0.97	0.97	0.97
D4	**0.95**	**0.95**	**0.95**	0.88	0.81	0.85	0.92	0.90	0.91	0.90	0.86	0.88
D5	**0.96**	**0.96**	**0.96**	0.87	0.74	0.80	5.92	0.89	0.91	0.89	0.85	0.87
D6	**0.99**	**0.99**	**0.99**	0.97	0.95	0.96	0.93	0.91	0.92	0.93	0.92	0.93
D7	**0.99**	**0.99**	**0.99**	0.90	0.80	0.85	0.79	0.81	0.80	0.94	0.88	0.91
H-mean.	**0.97**	**0.97**	**0.97**	0.92	0.80	0.85	0.89	0.88	0.89	0.93	0.92	0.92

candidates and identities. Compare with the most recent systems, SLINT highly outperforms the precision and recall in interlinking. The performance of SLINT is also very high when it takes around 1 minute to detect more than 13,000 identity pairs.

Although SLINT has good result on tested datasets, it is not sufficient to evaluate the scalability of our system, which we consider as the current limiting point because of the used of weighted co-occurrence matrix. We will investigate about a solution for this issue in our next work. Besides, we also interested in automatic configuration for every threshold used in SLINT and improving SLINT into a novel cross-domain interlinking system.

References

1. S. Araujo, D. Tran, A. de Vries, J. Hidders, and D. Schwabe. SERIMI: Class-based disambiguation for effective instance matching over heterogeneous web data. In *SIGMOD'12 15th Workshop on Web and Database*, pages 19–25, 2012.
2. I. F. Cruz, F. P. Antonelli, and C. Stroe. AgreementMaker: efficient matching for large real-world schemas and ontologies. *VLDB Endow.*, 2:1586–1589, 2009.
3. R. Isele and C. Bizer. Learning linkage rules using genetic programming. In *ISWC' 11 6th Workshop on Ontology Matching*, pages 13–24, 2011.
4. A. Ngomo, J. Lehmann, S. Auer, and K. Höffner. RAVEN - Active learning of link specifications. In *ISWC' 11 6th Workshop on Ontology Matching*, pages 25–36, 2011.
5. K. Nguyen, R. Ichise, and B. Le. Learning approach for domain-independent linked data instance matching. In *KDD'12 2nd Workshop on Minning Data Semantic*, pages 7:1–7:8, 2012.
6. A. Nikolov, M. d'Aquin, and E. Motta. Unsupervised learning of link discovery configuration. In *ESCW'12*, pages 119–133, 2012.
7. X. Niu, S. Rong, Y. Zhang, and H. Wang. Zhishi.links results for OAEI 2011. In *ISWC' 11 6th Workshop on Ontology Matching*, pages 220–227, 2011.
8. A. Schultz, A. Matteini, R. Isele, C. Bizer, and C. Becker. LDIF - Linked data integration framework. In *ISWC' 11 2nd Workshop on Consuming Linked Data*, 2011.
9. D. Song and J. Heflin. Automatically generating data linkages using a domain-independent candidate selection approach. In *ISWC' 11*, pages 649–664, 2011.
10. J. Volz, C. Bizez, M. Gaedke, and G. Kobilarov. Discovering and maintaining links on the web of data. In *ISWC' 09*, pages 650–665, 2009.

Learning Conformation Rules for Linked Data Integration

Axel-Cyrille Ngonga Ngomo[1]

Department of Computer Science
University of Leipzig
Johannisgasse 26, 04103 Leipzig
ngonga@informatik.uni-leipzig.de,
WWW home page: http://bis.uni-leipzig.de/AxelNgonga

Abstract. Over the last years, manifold applications that consume, link and integrate Linked Data have been developed. Yet, the specification of integration processes for Linked Data is rendered increasingly tedious by several factors such as the great number of knowledge bases in the Linked Data Cloud as well as schema mismatches and heterogeneous conventions for property values across knowledge bases. Especially the specification of rules for transforming property values has been carried out mostly manually so far. In this paper, we present CaRLA, an algorithm that allows learning transformation rules for pairs of property values expressed as strings. We present both a batch and an active learning version of CaRLA. The batch version of CaRLA uses a three-step learning approach to retrieve probable transformation rules. The active learning version extends the batch version by requesting highly informative property value pairs from the user so as to improve the learning speed of the system. We evaluate both versions of our approach on four experiments with respect to runtime and accuracy. Our results show that we can improve the precision of the data integration process by up to 12% by discovering transformation rules with human accuracy even when provided with small training datasets. In addition, we can even discover rules that were missed by human experts.

1 Introduction

The Linked Open Data (LOD) Cloud consists of more than 30 billion triples[1]. Making use of this large amount of domain knowledge within large-scale semantic applications is currently gaining considerable momentum. For example, the DeepQA framework [4] combines knowledge from DBpedia[2], Freebase[3] and several other knowledge bases to determine the answer to questions with a speed superior to that of human champions. Complex applications that rely on several sources of knowledge usually integrate them into a unified view by the means of

[1] http://www4.wiwiss.fu-berlin.de/lodcloud/state/
[2] http://dbpedia.org
[3] http://www.freebase.com

the Extract-Transform-Load (ETL) paradigm [7]. Yet, over the last few years, the automatic provision of such unified views on Linked datasets has been rendered increasingly tedious. The difficulties behind the integration of Linked Data are not only caused by the mere growth of the datasets in the Linked Data Web but also by large disparity across these datasets.

Linked Data Integration is commonly impeded by two categories of mismatches: ontology mismatches and naming convention mismatches. The second category of common mismatches (which is the focus of this work) mostly affects the transformation step of ETL and lies in the different conventions used for equivalent property values. For example, the labels of films in DBpedia differ from the labels of films in LinkedMDB[4] in three ways: First, they contain a language tag. Second, the extension "(film)" is added to the label of movies if another entity with the same label exists. Third, if another film with the same label exists, the production year of the film is added. Consequently, the film `Liberty` from 1929 has the label "Liberty (1929 film)@en" in DBpedia, while the same film bears the label "Liberty" in LinkedMDB. A similar discrepancy in naming persons holds for film directors and actors. Finding a conform representation of the labels of movies that maps the LinkedMDB representation would require knowing the rules `replace(''@en'', ϵ)` and `replace(''(*film)'', ϵ)` where ϵ stands for the empty string.

In this paper, we address the problem of discovering transformation rules by presenting CaRLA, the Canonical Representation Learning Algorithm. Our approach learns canonical (also called conform) representation of data type property values by implementing a simple, time-efficient and accurate learning approach. We present two versions of CaRLA: a batch learning and an active learning version. The batch learning approach relies on a training dataset to derive rules that can be used to generate conform representations of property values. The active version of CaRLA (aCarLa) extends CaRLA by computing unsure rules and retrieving highly informative candidates for annotation that allow the validation or negation of these candidates. One of the main advantages of CaRLA is that it can be configured to learn transformations at character, n-gram or even word level. By these means, it can be used to improve integration and link discovery processes based on string similarity/distance measures ranging from character-based (edit distance) and n-gram-based (q-grams) to word-based (Jaccard similarity) approaches.

The rest of this paper is structured as follows: In Section 2, we give an overview of the notation used in this paper. Thereafter, we present the two different versions of the CaRLA algorithm: the batch version in Section 3 and the active version in Section 4. Subsequently, we evaluate our approach with respect to both runtime and accuracy in four experiments (Section 5). After a brief overview of the related work (Section 6), we present some future work and conclude in Section 7.

[4] http://linkedmdb.org/

2 Preliminaries

In the following, we define terms and notation necessary to formalize the approach implemented by CaRLA. Let $s \in \Sigma^*$ be a string from an alphabet Σ. We define a tokenization function as follows:

Definition 1 (Tokenization Function). *Given an alphabet A of tokens, a tokenization function token : $\Sigma^* \to 2^A$ maps any string $s \in \Sigma^*$ to a subset of the token alphabet A.*

Note that string similarity and distances measures rely on a large number of different tokenization approaches. For example, the Levenshtein similarity [8] relies on a tokenization at character level, while the Jaccard similarity [6] relies on a tokenization at word level.

Definition 2 (Transformation Rule). *A transformation rule is a function $r : A \to A$ that maps a token from the alphabet A to another token of A.*

In the following we will denote transform rules by using an arrow notation. For example, the mapping of the token "`Alan`" to "`A.`" will be denoted by $<$ "`Alan`" $\to$ "`A.`"$>$. For any rule $r =< x \to y >$, we call x the *premise* and y the *consequence* of r. We call a transformation rule *trivial* when it is of the form $<$ `x` $\to$ `x` $>$ with $x \in A$. We call two transformation rules r and r' *inverse* to each other when $r =<$ `x` $\to$ `y` $>$ and $r' =<$ `y` $\to$ `x` $>$. Throughout this work, we will assume that the characters that make up the tokens of A belong to $\Sigma \cup \{\epsilon\}$, where ϵ stands for the empty character. Note that we will consequently denote deletions by rules of the form $< x \to \epsilon >$ where $x \in A$.

Definition 3 (Weighted Transformation Rule). *Let Γ be the set of all rules. Given a weight function $w : \Gamma \to \mathbb{R}$, a weighted transformation rule is the pair $(r, w(r))$, where $r \in \Gamma$ is a transformation rule.*

Definition 4 (Transformation Function). *Given a set R of (weighted) transformation rules and a string s, we call the function $\varphi_R : \Sigma^* \to \Sigma * \cup \{\epsilon\}$ a transformation function when it maps s to a string $\varphi_R(s)$ by applying all rules $r_i \in R$ to every token of token(s) in an arbitrary order.*

For example, the set $R = \{<$"`Alan`"$\to$"`A.`"$>\}$ of transformation rules would lead to φ_R("James Alan Hetfield") ="James A. Hetfield".

3 Batch Learning Approach

The goal of CaRLA is two-fold: First, it aims to compute rules that allow the derivation of conform representations of property values. As entities can have several values for the same property, CaRLA also aims to detect a condition under which two property values should be merged during the integration process. In the following, we will assume that two source knowledge bases are to be integrated to one. Note that our approach can be used for any number of source knowledge bases.

3.1 Overview

Formally, CaRLA addresses the problem of finding the required transformation rules by computing an equivalence relation $\mathcal{E}$ between pairs of property values (p_1, p_2) that is such that $\mathcal{E}(p_1, p_2)$ holds when p_1 and p_2 should be mapped to the same *canonical representation* p. CaRLA computes $\mathcal{E}$ by generating two sets of weighted transformation function rules R_1 and R_2 such that for a given similarity function σ, $\mathcal{E}(p_1, p_2) \rightarrow \sigma(\varphi_{R_1}(p_1), \varphi_{R_2}(p_2)) \geq \theta$, where θ is a similarity threshold. The canonical representation p is then set to $\varphi_{R_1}(p_1)$. The similarity condition $\sigma(\varphi_{R_1}(p_{R_1}), \varphi_{R_2}(p_2)) \geq \theta$ is used to distinguish between the pairs of properties values that should be merged.

To detect R_1 and R_2, CaRLA assumes two training datasets P and N, of which N can be empty. The set P of positive training examples is composed of pairs of property value pairs (p_1, p_2) such that $\mathcal{E}(p_1, p_2)$ holds. The set N of negative training examples consists of pairs (p_1, p_2) such that $\mathcal{E}(p_1, p_2)$ does not hold. In addition, CaRLA assumes being given a similarity function σ and a corresponding tokenization function *token*. Given this input, CaRLA implements a three-step approach. It begins by computing the two sets R_1 and R_2 of plausible transformation rules based on the positive examples at hand (Step 1). Then it merges inverse rules across R_1 and R_2 and discards rules with a low weight during the rule merging and filtering step. From the resulting set of rules, CaRLA derives the similarity condition $\mathcal{E}(p_1, p_2) \rightarrow \sigma(\varphi_{R_1}(p_1), \varphi_{R_2}(p_2)) \geq \theta$. It then applies these rules to the negative examples in N and tests whether the similarity condition also holds for the negative examples. If this is the case, then it discards rules until it reaches a local minimum of its error function. The retrieved set of rules and the novel value of θ constitute the output of CaRLA and can be used to generate the canonical representation of the properties in the source knowledge bases.

In the following, we explain each of the three steps in more detail. Throughout the explanation we use the toy example shown in Table 1. In addition, we will assume a word-level tokenization function and the Jaccard similarity.

Type	Property value 1	Property value 2
$\oplus$	"Jean van Damne"	"Jean Van Damne (actor)"
$\oplus$	"Thomas T. van Nguyen"	"Thomas Van Nguyen (actor)"
$\oplus$	"Alain Delon"	"Alain Delon (actor)"
$\oplus$	"Alain Delon Jr."	"Alain Delon Jr. (actor)"
$\ominus$	"Claude T. Francois"	"Claude Francois (actor)"

Table 1. Toy example dataset. The positive examples are of type $\oplus$ and the negative of type $\ominus$.

3.2 Rule Generation

The goal of the rule generation set is to compute two sets of rules R_1 resp. R_2 that will underlie the transformation φ_{R_1} resp. φ_{R_2}. We begin by tokenizing all positive property values p_i and p_j such that $(p_i, p_j) \in P$. We call T_1 the set of all tokens p_i such that $(p_i, p_j) \in P$, while T_2 stands for the set of all p_j. We begin the computation of R_1 by extending the set of tokens of each $p_j \in T_2$ by adding ϵ to it. Thereafter, we compute the following rule score function *score*:

$$score(< x \to y >) = |\{(p_i, p_j) \in P : x \in token(p_i) \wedge y \in token(p_j)\}|. \quad (1)$$

score computes the number of co-occurrences of the tokens x and y across P. All tokens $x \in T_1$ always have a maximal co-occurrence with ϵ as it occurs in all tokens of T_2. To ensure that we do not only compute deletions, we decrease the score of rules $< x \to \epsilon >$ by a factor $\kappa \in [0, 1]$. Moreover, in case of a tie, we assume the rule $< x \to y >$ to be more natural than $< x \to y' >$ if $\sigma(x, y) > \sigma(x, y')$. Given that σ is bound between 0 and 1, it is sufficient to add a fraction of $\sigma(x, y)$ to each rule $< x \to y >$ to ensure that the better rule is chosen. Our final score function is thus given by

$$score_{final}(< x \to y >) = \begin{cases} score(< x \to y >) + \sigma(x, y)/2. & \text{if } y \neq \epsilon, \\ \kappa \times score(< x \to y >) & \text{else.} \end{cases} \quad (2)$$

Finally, for each token $x \in T_1$, we add the rule $r =< x \to y >$ to R_1 iff $x \neq y$ (i.e., r is not trivial) and $y = \arg\max_{y' \in T_2} score_{final}(< x \to y' >)$. To compute R_2 we simply swap T_1 and T_2, invert P (i.e., compute the set $\{(p_j, p_i) : (p_i, p_j) \in P\}$) and run through the procedure described above.

For the set P in our example, we get the following sets of rules: $R_1 = \{(<$"van"$\to$"Van"$>$, 2.08), $(<$"T."$\to \epsilon >$, 2)$\}$ and $R_2 = \{(<$"Van"$\to$"van"$>$, 2.08), $(<$"(actor)"$\to \epsilon >$, 2)$\}$.

3.3 Rule Merging and Filtering

The computation of R_1 and R_2 can lead to a large number of inverse or improbable rules. In our example, R_1 contains the rule $<$"van"$\to$"Van"$>$ while R_2 contains $<$"Van"$\to$"van"$>$. Applying these rules to the data would consequently not improve the convergence of their representations. To ensure that the transformation rules lead to similar canonical forms, the rule merging step first discards all rules $< x \to y > \in R_2$ that are such that $< y \to x > \in R_1$ (i.e., rules in R_2 that are inverse to rules in R_1). Then, low-weight rules are discarded. The idea here is that if there is not enough evidence for a rule, it might just be a random event. The initial similarity threshold θ for the similarity condition is finally set to

$$\theta = \min_{(p_1, p_2) \in P} \sigma(\varphi_{R_1}(p_1), \varphi_{R_2}(p_2)). \quad (3)$$

In our example, CaRLA would discard $<$"van"$\to$"Van"$>$ from R_2. When assuming a threshold of 10% of P's size (i.e., 0.4), no rule would be filtered out.

The output of this step would consequently be $R_1 = \{(<\text{“van”}\rightarrow\text{“Van”}>, 2.08), (<\text{“T.”}\rightarrow \epsilon >, 2)\}$ and $R_2 = \{(<\text{“(actor)”}\rightarrow \epsilon >, 2)\}$.

3.4 Rule Falsification

The aim to the rule falsification step is to detect a set of transformations that lead to a minimal number of elements of N having a similarity superior to θ via σ. To achieve this goal, we follow a greedy approach that aims to minimize the magnitude of the set

$$E = \{(p_1, p_2) \in N : \sigma(\varphi_{R_1}(p_1), \varphi_{R_2}(p_2)) \geq \theta = \min_{(p_1,p_2)\in P} \sigma(\varphi_{R_1}(p_1), \varphi_{R_2}(p_2))\}. \tag{4}$$

Our approach simply tries to discard all rules that apply to elements of E by ascending score. If E is then empty, the approach terminates. If E does not get smaller, then the change is rolled back and then the next rule is tried. Else, the rule is discarded from the set of final rules. Note that discarding a rule can alter the value of θ and thus E. Once the set E has been computed, CaRLA concludes its computation by generating a final value of the threshold θ.

In our example, two rules apply to the element of N. After discarding the rule $<\text{“T.”}\rightarrow \epsilon >$, the set E becomes empty, leading to the termination of the rule falsification step. The final set of rules are thus $R_1 = \{<\text{“van”}\rightarrow\text{“Van”}>\}$ and $R_2 = \{<\text{“(actor)”}\rightarrow \epsilon >\}$. The value of θ is computed to be 0.75. Table 2 shows the canonical property values for our toy example. Note that this threshold allows to discard the elements of N as being equivalent property values.

Property value 1	Property value 2	Canonical value
“Jean van Damne”	“Jean Van Damne (actor)”	“Jean Van Damne”
“Thomas T. van Nguyen”	“Thomas Van Nguyen (actor)”	“Thomas T. Van Nguyen”
“Alain Delon”	“Alain Delon (actor)”	“Alain Delon”
“Alain Delon Jr.”	“Alain Delon Jr. (actor)”	“Alain Delon Jr.”
“Claude T. Francois”		“Claude T. Francois”
	“Claude Francois (actor)”	“Claude Francois”

Table 2. Canonical property values for our example dataset

It is noteworthy that by learning transformation rules, we also found an initial threshold θ for determining the similarity of property values using σ as similarity function. In combination with the canonical forms computed by CaRLA, the configuration (σ, θ) can be used as an initial configuration for Link Discovery frameworks such as LIMES. For example, the smallest Jaccard similarity for the pair of property values for our example lies by 1/3, leading to a precision of 0.71 for a recall of 1 (F-measure: 0.83). Yet, after the computation of the transformation rules, we reach an F-measure of 1 with a threshold of 1. Consequently, the pair (σ, θ) can be used for determining an initial classifier for approaches such as the RAVEN [11] algorithm implemented in LIMES [10].

4 Extension to Active Learning

One of the drawbacks of batch learning approaches is that they often require a large number of examples to generate good models. As our evaluation shows (see Section 5), this drawback also holds for the batch version of CaRLA, as it can easily detect very common rules but sometimes fails to detect rules that apply to less pairs of property values. In the following, we present how this problem can be addressed by extending CaRLA to aCARLA using active learning [16].

Algorithm 1 Overview of aCaRLA

Require: Positive examples P_0
Require: Negative examples N_0
Require: Similarity function σ
Require: Damping factor κ
Require: Score threshold s_{min}
Require: Tokenization function *token*
Require: Maximal number of annotation requests q_{total}
Require: Number of questions/iteration q
  Rule sets $R_1 \leftarrow \emptyset$, $R_2 \leftarrow \emptyset$
  $q_{current} := 0$ //Current number of questions
  $Ex := 0$ //Set of examples to annotate
  $t := 0$ //Iteration counter
  $r := null$ //unsure rule
  $B := \emptyset$ //set of banned rules
  while $q_{current} \leq q_{total}$ **do**
    (R_1, R_2, θ) = runCarla(P_t, N_t, σ, κ, S_{min}, *token*) // Run batch learning
    r =getMostUnsureRule($R_1 \cup R_2$, B, s_min)
    if $r \neq null$ **then**
      Ex = computeExamples(r, q)
    else
      Ex = getMostDifferent(q)
    end if
    (P, N) = requestAnnotations(Ex)
    $P_{t+1} \leftarrow P_t \cup P$
    $N_{t+1} \leftarrow N_t \cup N$
    $B \leftarrow$ updateBannedRules(B, r)
    $t \leftarrow t + 1$
    $q_{current} \leftarrow q_{current} + |Ex|$
  end while
  (R_1, R_2, θ) := runCarla(P_t, N_t, σ, κ, S_{min}, *token*)

  return (R_1, R_2, θ)

An overview of aCaRLA is given in Algorithm 1. The basic idea here is to begin with small training sets P_0 and N_0. In each iteration, all the available training data is used by the batch version of CaRLA to update the set of rules.

The algorithm then tries to refute or validate rules with a score below the score threshold s_{min} (i.e., unsure rules). For this purpose it picks the most unsure rule r that has not been shown to be erroneous in a previous iteration (i.e., that is not an element of the set of banned rules B). It then fetches a set Ex of property values that map the left side (i.e., the premise) of r. Should there be no unsure rule, then Ex is set to the q property values that are most dissimilar to the already known property values. Annotations consisting of the corresponding values for the elements of Ex in the other source knowledge bases are requested by the user and written in the set P. Property values with no corresponding values are written in N. Finally the sets of positive and negative examples are updated and the triple (R_1, R_2, θ) is learned anew until a stopping condition such as a maximal number of questions is reached. As our evaluation shows, this simple extension of the CaRLA algorithm allows it to detect efficiently the pairs of annotations that might lead to a larger set of high-quality rules.

5 Evaluation

5.1 Experimental Setup

In the experiments reported in this section, we evaluated CaRLA by two means: First we aimed to measure how well CaRLA could compute transformations created by experts. To achieve this goal, we retrieved transformation rules from four link specifications defined manually by experts within the LATC project[5]. An overview of these specifications is given in Table 3. Each link specification aimed to compute `owl:sameAs` links between entities across two knowledge bases by first transforming their property values and by then computing the similarity of the entities based on the similarity of their property values. For example, the computation of links between films in DBpedia and LinkedMDB was carried out by first applying the set of $R_1 = \{< (film) \rightarrow \epsilon >\}$ to the labels of films in DBpedia and $R_2 = \{< (director) \rightarrow \epsilon >\}$ to the labels of their directors. We ran both CaRLA and aCaRLA on the property values of the interlinked entities and measured how fast CaRLA was able to reconstruct the set of rules that were used during the Link Discovery process.

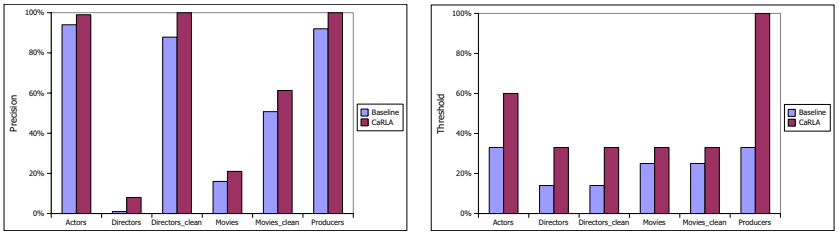
Experiment	Source	Target	Source property	Target property	Size
Actors	DBpedia	LinkedMDB	`rdfs:label`	`rdfs:label`	1172
Directors	DBpedia	LinkedMDB	`rdfs:label`	`rdfs:label`	7353
Movies	DBpedia	LinkedMDB	`rdfs:label`	`rdfs:label`	9859
Producers	DBpedia	LinkedMDB	`rdfs:label`	`rdfs:label`	1540

Table 3. Overview of the datasets

In addition, we quantified the quality of the rules learned by CaRLA. In each experiments, we computed the boost in the precision of the mapping of property pairs with and without the rules derived by CaRLA. The initial precision was

[5] http://latc-project.eu

computed as $\frac{|P|}{|M|}$, where $M = \{(p_i, p_j) : \sigma(p_i, p_j) \geq \min_{(p_1,p_2)\in P} \sigma(p_1, p_2)\}$. The precision after applying CaRLA's results was computed as $\frac{|P|}{|M'|}$ where $M' = \{(p_i, p_j) : \sigma(\varphi_{R_1}(p_i), \varphi_{R_2}(p_j)) \geq \min_{(p_1,p_2)\in P} \sigma(\varphi_{R_1}(p_1), \varphi_{R_2}(p_2))\}$. Note that in both cases, the recall was 1 given that $\forall (p_i, p_j) \in P : \sigma(p_i, p_j) \geq \min_{(p_1,p_2)\in P} \sigma(p_1, p_2)$. In all experiments, we used the Jaccard similarity metric and a word tokenizer with $\kappa = 0.8$. All runs were carried on a notebook running Windows 7 Enterprise with 3GB RAM and an Intel Dual Core 2.2GHz processor. Each of the algorithms was ran five times. We report the rules that were discovered by the algorithms and the number of experiments within which they were found.



(a) Precision (b) Thresholds

Fig. 1. Comparison of the precision and thresholds with and without CaRLA.

5.2 Results and Discussion

Table 4 shows the union of the rules learned by the batch version of CaRLA in all five runs. Note that the computation of a rule set lasted under 0.5s even for the largest dataset, Movies. The columns P_n give the probability of finding a rule for a training set of size n in our experiments. R_2 is not reported because it was empty in all setups. Our results show that in all cases, CaRLA converges quickly and learns rules that are equivalent to those utilized by the LATC experts with a sample set of 5 pairs. Note that for each rule of the form $<$"@en" $\rightarrow y>$ with $y \neq \epsilon$ that we learned, the experts used the rule $< y \rightarrow \epsilon >$ while the linking platform automatically removed the language tag. We experimented with the same datasets without language tags and computed exactly the same rules as those devised by the experts. In some experiments (such as Directors), CaRLA was even able detect rules that where not included in the set of rules generated by human experts. For example, the rule $<$"(filmmaker)" $\rightarrow$ "(director)"$>$ is not very frequent and was thus overlooked by the experts. In Table 4, we marked such rules with an asterisk. The director and the movies datasets contained a large number of typographic errors of different sort (incl. misplaced hyphens, character repetitions such as in the token "Neilll", etc.) which led to poor precision scores in our experiments. We cleaned the first 250 entries of these datasets

from these errors and obtained the results in the rows labels Directors_clean and Movies_clean. The results of CaRLA on these datasets are also shown in Table 4. We also measured the improvement in precision that resulted from applying CaRLA to the datasets at hand (see Figure 1). For that the precision remained constant across the different dataset sizes. In the best case (cleaned Directors dataset), we are able to improve the precision of the property mapping by 12.16%. Note that we can improve the precision of the mapping of property values even on the noisy datasets.

Experiment	R_1	P_5	P_{10}	P_{20}	P_{50}	P_{100}
Actors	$<$"@en" $\rightarrow$ "(actor)" $>$	1	1	1	1	1
Directors	$<$"@en" $\rightarrow$ "(director)" $>$	1	1	1	1	1
Directors	$<$"(filmmaker)" $\rightarrow$ "(director)"$>^*$	0	0	0	0	0.2
Directors_clean	$<$"@en" $\rightarrow$ "(director)" $>$	1	1	1	1	1
Movies	$<$"@en" $\rightarrow \epsilon >$	1	1	1	1	1
Movies	$<$"(film)" $\rightarrow \epsilon >$	1	1	1	1	1
Movies	$<$"film)" $\rightarrow \epsilon >^*$	0	0	0	0	0.6
Movies_clean	$<$"@en" $\rightarrow \epsilon >$	1	1	1	1	1
Movies_clean	$<$"(film)" $\rightarrow \epsilon >$	0	0.8	1	1	1
Movies_clean	$<$"film)" $\rightarrow \epsilon >^*$	0	0	0	0	1
Producers	$<$"@en" $\rightarrow$ (producer)$>$	1	1	1	1	1

Table 4. Overview of batch learning results

Interestingly, when used on the Movies dataset with a training dataset size of 100, our framework learned low-confidence rules such as $<$"(1999" $\rightarrow \epsilon >$, which were yet discarded due to a too low score. These are the cases where aCaRLA displayed its superiority. Thanks to its ability to ask for annotation when faced with unsure rules, aCaRLA is able to validate or negate unsure rules. As the results on the Movies example show, aCaRLA is able to detect several supplementary rules that were overlooked by human experts. Especially, it clearly shows that deleting the year of creation of a movie can improve the conformation process. aCaRLA is also able to generate a significantly larger number of candidates rules for the user's convenience.

6 Related Work

Linked Data Integration is an important topic for all applications that rely on a large number of knowledge bases and necessitate a unified view on this data, e.g., Question Answering frameworks [4] and semantic mashups [13]. In recent work, several challenges and requirements to Linked Data consumption and integration have been pointed out [9]. Recently, several approaches and frameworks have been developed with the aim of addressing many of these challenges. For example, the R2R framework [2] allows the specification and publication of mappings

Experiment	R_1	P_5	P_{10}	P_{20}	P_{50}	P_{100}
Actors	<"@en" → "(actor)" >	1	1	1	1	1
Directors	<"@en" → "(director)" >	1	1	1	1	1
Directors	<"(actor)" → "(director)">*	0	0	0	0	1
Directors_clean	<"@en" → "(director)" >	1	1	1	1	1
Movies	<"@en" → ϵ >	1	1	1	1	1
Movies	<"(film)" → ϵ >	1	1	1	1	1
Movies	<"film)" → ϵ >*	0	0	0	0	1
Movies	<"(2006" → ϵ >*	0	0	0	0	1
Movies	<"(199" → ϵ >*	0	0	0	0	1
Movies_clean	<"@en" → ϵ >	1	1	1	1	1
Movies_clean	<"(film)" → ϵ >	0	1	1	1	1
Movies_clean	<"film)" → ϵ >*	0	0	0	0	1
Producers	<"@en" → (producer)>	1	1	1	1	1

Table 5. Overview of active learning results

that allow mapping resources and literals across knowledge bases. The Linked Data Integration Framework LDIF [15], whose goal is to support the integration of RDF data, builds upon R2R mappings and technologies such as SILK [5] and LDSpider[6]. The concept behind the framework is to enable users to create periodic integration jobs via simple XML configurations. KnoFuss [12] addresses data integration from the point of view of link discovery by monitoring the interaction between instance and dataset matching (which is similar to ontology matching [3]). Also worth mentioning id Semantic Web Pipes[7] [13], which follows the idea of Yahoo Pipes[8] to enable the integration of data in formats such as RDF and XML. In all of these systems, the configuration of the data transformations has to be carried out manually.

Some approaches to transformation rule learning approaches have previously been proposed in the record linkage area. For example, [14] presents an interactive approach to data cleaning while [1] developed an approach to learn string transformation from examples. Yet, none of these approaches uses active learning to improve the number of rules it detects. To the best of our knowledge, CaRLA is the first approach tailored towards Linked Data that allows the active learning of data conformation rules for property values expressed as strings.

7 Conclusion and Future Work

In this work, we presented two algorithms for learning data transformations. We present a batch learning approach that allows to detect rules by performing a co-occurrence analysis. This version is particularly useful when the knowledge bases to be integrated are already linked. We also present an active learning approach

[6] http://code.google.com/p/ldspider/
[7] http://pipes.deri.org/
[8] http://pipes.yahoo.com/pipes/

that allows to discover a large number of rules efficiently. We evaluated our approach with respect to the quality of the rules it discovers and to the effect of the rules on matching property values. We showed that the data transformation achieved by our approach allows to increase the precision of property mapping by up to 12% when the recall is set to 1. In future work, we aim to extend our approach to learning transformation rules with several tokenizers concurrently.

References

1. Arvind Arasu, Surajit Chaudhuri, and Raghav Kaushik. Learning string transformations from examples. *Proc. VLDB Endow.*, 2(1):514–525, August 2009.
2. Christian Bizer and Andreas Schultz. The R2R Framework: Publishing and Discovering Mappings on the Web. In *Proceedings of COLD*, 2010.
3. Jérôme Euzenat and Pavel Shvaiko. *Ontology matching*. Springer-Verlag, Heidelberg (DE), 2007.
4. David A. Ferrucci, Eric W. Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John M. Prager, Nico Schlaefer, and Christopher A. Welty. Building watson: An overview of the deepqa project. *AI Magazine*, 31(3):59–79, 2010.
5. Robert Isele and Christian Bizer. Learning Linkage Rules using Genetic Programming. In *Sixth International Ontology Matching Workshop*, 2011.
6. Paul Jaccard. Étude comparative de la distribution florale dans une portion des Alpes et des Jura. *Bulletin del la Société Vaudoise des Sciences Naturelles*, 37:547–579, 1901.
7. Ralph Kimball and Joe Caserta. *The Data Warehouse ETL Toolkit: Practical Techniques for Extracting, Cleaning, Conforming, and Delivering Data*. Wiley, 2004.
8. Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. Technical Report 8, 1966.
9. Ian Millard, Hugh Glaser, Manuel Salvadores, and Nigel Shadbolt. Consuming multiple linked data sources: Challenges and experiences. In *First International Workshop on Consuming Linked Data*, 2010.
10. Axel-Cyrille Ngonga Ngomo. A Time-Efficient Hybrid Approach to Link Discovery. In *Sixth International Ontology Matching Workshop*, 2011.
11. Axel-Cyrille Ngonga Ngomo, Jens Lehmann, Sören Auer, and Konrad Höffner. RAVEN – Active Learning of Link Specifications. In *Proceedings of OM@ISWC*, 2011.
12. Andriy Nikolov, Victoria Uren, Enrico Motta, and Anne Roeck. Overcoming schema heterogeneity between linked semantic repositories to improve coreference resolution. In *Proceedings of the 4th Asian Conference on The Semantic Web*, pages 332–346, 2009.
13. Danh Le Phuoc, Axel Polleres, Manfred Hauswirth, Giovanni Tummarello, and Christian Morbidoni. Rapid prototyping of semantic mash-ups through semantic web pipes. In *WWW*, pages 581–590, 2009.
14. Vijayshankar Raman and Joseph M. Hellerstein. Potter's wheel: An interactive data cleaning system. In *VLDB*, pages 381–390, 2001.
15. Andreas Schultz, Andrea Matteini, Robert Isele, Christian Bizer, and Christian Becker. Ldif - linked data integration framework. In *COLD*, 2011.
16. Burr Settles. Active learning literature survey. Technical Report 1648, University of Wisconsin-Madison, 2009.

Coupling of WordNet Entries for Ontology Mapping using Virtual Documents

Frederik C. Schadd and Nico Roos

Department of Knowledge Engineering,
Maastricht University, Maastricht, The Netherlands,
frederik.schadd@maastrichtuniversity.nl,
roos@maastrichtuniversity.nl

Abstract.
Facilitating information exchange is a crucial service for ontology-based knowledge systems. This can be achieved by the mapping of two heterogenous ontologies. Many mapping frameworks utilize language-based knowledge resources such as WordNet. By coupling all ontology concepts to a corresponding entry in WordNet, one can quantify the lexical relatedness of any two ontology concepts. However, coupling the correct entry is a difficult task due to the ambiguous nature of names. Coupling the wrong entries hence yields similarity values that do not correctly express the relatedness of two given concepts, resulting in a poor performance of the overall mapping framework. This paper proposes an approach for the more accurate coupling of ontology concepts with their corresponding WordNet entries. The basis of the proposed approach is the creation of separate virtual documents representing the different ontology concepts and WordNet entries and coupling these according to their document similarities. The extent of improvements using this approach are evaluated using a data set originating from the Ontology Alignment Evaluation Initiative (OAEI). Furthermore, the performance of a framework using our approach is demonstrated using the results of the OAEI 2011 competition.

1 Introduction

Sharing and reusing knowledge is an important aspect in modern information systems. Since multiple decades, researchers have been investigating methods that facilitate knowledge sharing in the corporate domain, allowing for instance the integration of external data into a company's own knowledge system. Ontologies are at the center of this research, allowing the explicit definition of a knowledge domain. With the steady development of ontology languages, such as the current OWL language [11], knowledge domains can be modelled with an increasing amount of detail. Due to the Semantic Web vision [2], information sources on the future World Wide Web will store machine readable information, allowing autonomous agents to collect and interpret information automatically. Just as in current knowledge systems, each information source on the World Wide Web will store its structured content with a publicly available ontology describing the semantics of stored information. Such ontologies are generally developed independently, resulting in many different ontologies describing the same

domain. Thus, agents roaming the Semantic Web need to be able to integrate knowledge of heterogenous sources into their own representation of a specific domain.

Commonly, ontology mapping tools combine a variety of similarity measures using advanced aggregation techniques. The application of an extraction technique on the aggregated similarities can then be used to produce an alignment. The focus of this article lies on similarities measures that utilize lexical ontologies. More specifically, we investigate the automatic identification of corresponding entries in these ontologies through the use of virtual documents and information retrieval techniques, such that the semantic relatedness of any two ontology entities can be accurately specified. This article expands on previous research [17] by applying a formal virtual document model and evaluating the system against state-of-the-art frameworks in the OAEI 2011 campaign.

2 Related Work

Matching heterogenous ontologies has traditionally been done either manually or using semi-automatic tools. However, many research groups have focused their attention on automatic mapping approaches. This has led to the development of ontology mapping frameworks [18] which all utilize different techniques and resources. Many of these include lexical ontologies such as WordNet in their matching procedure. Falcon-AO [15] was the first framework to successfully apply the concept of virtual documents in the ontology mapping process. Here, virtual documents are created where each document represents a different ontology concept, such that a similarity matrix can be computed by applying a document similarity measure on the virtual documents.

Budanitsky et al. [3] evaluated five different measures of expressing the semantic relatedness between WordNet concepts, which subsequently can be applied to approaches that use different lexical ontologies. Buitelaar et al. [4] proposed a linguistic model as a labelling system, such that natural language can be generated using the ontology concepts. Such a model would be useful for situations when the name of a concept has no matching entry in a lexical ontology, allowing the linguistic decomposition of a name such that an appropriate lexical entry might still be mapped.

The techniques applied in this research are related to the field of Word Sense Disambiguation, which can be approached using numerous different techniques [14]. The strongest related technique is the *Lesk* [10] method, however key differences to the proposed approach is that it is limited to the glossary of the concept, omitting other information such as labels and the data of related concepts, and that it does not allow for a weighting of the terms according to a specified document model. The *Extended-Lesk* [1] method does also incorporate glossaries of related concepts, however still lacks the inclusion of non-glossary information and the weighting of terms according to their origin within the ontology, which the proposed approach does provide.

3 Motivation

Lexical ontologies are useful assets for ontology mapping systems. Established research primarily focused on developing frameworks or theoretical models which allow sophisticated reasoning functionalities, provided the ontology concepts are annotated, or

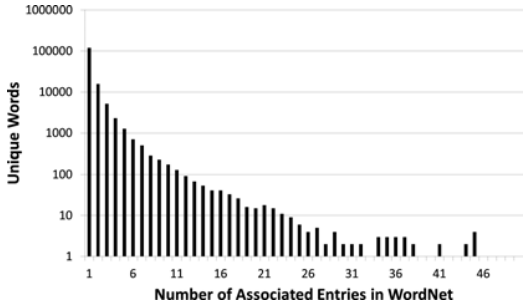


Fig. 1. Frequency of the amounts of possible concepts that can be coupled to a given word. All unique words occurring in WordNet were used as data.

'coupled', using the framework constructs. However, in order to utilize a lexical ontology or appropriate framework it is necessary that the given ontologies actually contain these couplings. Unfortunately, this is rarely the case, meaning that the ontology concepts need to be coupled during the mapping procedure. This is not a straight-forward task since words can have different meaning, such that when looking up the name of a concept in a lexical ontology it can occur that there are multiple entries for that name. Figure 1 indicates the extent of such situations occurring within WordNet [13], by displaying the frequency of the amounts of possible concepts that can be coupled to a given word, using all unique concept labels that occur in WordNet as queries.

From Figure 1 one can see that, while there is a large collection of words that only have one entry in WordNet, a significant proportion of the data leads to multiple entries. This issue becomes increasingly prevalent when the concept names do not directly occur in a lexical ontology, due to the names being composite words or technical terms. Research is required into methods that can automatically couple ontology concepts to entries in lexical ontologies for the situation when such couplings are not specified.

4 Virtual Documents

We will provide a brief introduction to virtual documents and provide a detailed description of the creation of a virtual document representing the meaning of a ontology concept or WordNet entry. The general definition of a virtual document [20] is any document for which no persistent state exists, such that some or all instances of the given document are generated at run-time. A simple example would be creating a template for a document and completing the document using values stored in a database.

In this domain the basic data structure used for the creation of a virtual document is a linked-data model. It consists of different types of binary relations that relate concepts in order to create an exploitable structure, i.e. a graph. RDF [9] is an example of a linked-data model, which can be used to denote an ontology according to the OWL specification [11]. The inherent data model of WordNet has similar capacities, however it stores its data using a database. A key feature of a linked-data model is that it not only

allows the extraction of literal data for a given concept, but also enables the exploration of concepts that are related to that particular concept, such that the information of these neighboring concepts can then be included in the virtual document.

We will provide a generalized description of the creation of a virtual document based on established research [15]. The generalization has the purpose of providing a description that is not only applicable to an OWL/RDF ontology like the description given in Qu et al. [15], but also to the WordNet model. To provide the functions that are used to create a virtual document, the following terminology is used:

Synset: Basic element within WordNet, used to denote a specific meaning using a list of synonyms. Synsets are related to other synsets by different semantic relations, such as hyponymy and holonymy.

Concept: A named entity in the linked-data model. A concept denotes a named class or property given an ontology and a synset when referring to WordNet.

Link: A basic component of a linked-data model for relating elements. A link is directed, originating from a source and pointing towards a target, such that the type of the link indicates what relation holds between the two elements. An example of a link is a triplet in an RDF graph.

sou(s), type(s), tar(s): The source element, type and target element of a link s, respectively. Within the RDF model, these three elements of a link are also known as the subject, predicate and object of a triplet.

Collection of words: A list of unique words where each word has a corresponding weight in the form of a rational number.

+: Operator denoting the merging of two collections of words.

A concept definition within a linked-data model contains different types of literal data, such as a name, different labels, annotations and comments. The RDF model expresses some of these values using the *rdfs:label*, *rdfs:comment* relations. Concept descriptions in WordNet have similar capacities, but the labels of a concepts are referred to as its synonyms and the comments of a concept are linked via the glossary relation.

Definition 1. *Let ω be a concept of a linked-data model, the description of ω is a collection of words defined by (1):*

$$\begin{aligned} Des(e) = \alpha_1 * &\textit{ collection of words in the name of } \omega \\ +\alpha_2 * &\textit{ collection of words in the labels of } \omega \\ +\alpha_3 * &\textit{ collection of words in the comments of } \omega \\ +\alpha_4 * &\textit{ collection of words in the annotations of } \omega \end{aligned} \tag{1}$$

α_1, α_2, α_3 and α_4 are each rational numbers in $[0, 1]$, such that words can be weighed according to their origin.

Next to accumulating information that is directly related to a specific concept, one can also include the descriptions of neighboring concepts that are associated with that concept via a link. Such a link can be a standard relation that is defined in the linked-data model, for instance the specialization relation. However, it can also be a relation that is defined specifically for this ontology, such as an object property in the OWL language. The OWL language supports the inclusion of blank-node concepts which allow complex

logical expressions to be included in concept definitions. However, since not all linked-data models support the blank-node functionality, among which WordNet, these are omitted in our generalization. For more information on how to include blank nodes in the description, consult the work by Qu et al. [15].

To explore neighboring concepts, three neighbor operations are defined. $SON(\omega)$ denotes the set of concepts that occur in any link for which ω is the source of that link. Likewise $TYN(\omega)$ denotes the set of concepts that occur in any link for which ω is the type of that link and $TAN(\omega)$ denotes the set of concepts that occur in any link for which ω is the target. WordNet contains inverse relations, such as hypernym being the inverse of the hyponym relation. When faced with two relations which are the inverse of each other, only one of the two should be used such that descriptions of neighbors are not included twice in the virtual document. The formal definition of the neighbor operators is given below.

Definition 2. *Let ω be a named concept and s be a variable representing an arbitrary link. The set of source neighbors $SON(\omega)$ is defined by (2), the set of type neighbors of ω is defined by (3) and the set of target neighbors of ω is defined by (4).*

$$SON(\omega) = \bigcup_{sou(s)=\omega} \{type(s), tar(s)\} \tag{2}$$

$$TYN(\omega) = \bigcup_{type(s)=\omega} \{sou(s), tar(s)\} \tag{3}$$

$$TAN(\omega) = \bigcup_{tar(s)=\omega} \{sou(s), type(s)\} \tag{4}$$

Given the previous definitions, the definition of a virtual document of a specific concept can be formulated as follows.

Definition 3. *Let ω be a concept of a linked-data model. The virtual document of ω, denoted as $VD(\omega)$, is defined by (5):*

$$\begin{aligned} VD(\omega) =& Des(\omega) + \beta_1 * \sum_{\omega' \in SON(\omega)} Des(\omega') \\ &+ \beta_2 * \sum_{\omega' \in TYN(\omega)} Des(\omega') + \beta_3 * \sum_{\omega' \in TAN(\omega)} Des(\omega') \end{aligned} \tag{5}$$

Here, β_1, β_2 and β_3 are rational numbers in $[0, 1]$. This makes it possible to allocate a different weight to the descriptions of neighboring concepts of ω compared to the description of the concept ω itself.

5 Coupling Synsets

Our proposed approach aims at improving matchers applying lexical ontologies, in this case WordNet. When applying WordNet for ontology mapping, one is presented with the problem of identifying the correct meaning, or synset, for each entity in both ontologies that are to be matched. The goal of our approach is to automatically identify

the correct synsets for each entity of an ontology using information retrieval techniques. Given two ontologies O_1 and O_2 that are to be matched, O_1 contains the sets of entities $E_x^1 = \{e_1^1, e_2^1, ..., e_m^1\}$, where x distinguishes between the set of classes, properties or instances, O_2 contains the sets of entities $E_x^2 = \{e_1^2, e_2^2, ..., e_n^2\}$, and $C(e)$ denotes a collection of synsets representing entity e. The main steps of our approach, performed separately for classes, properties and instances, can be described as follows:

1. For every entity e in E_x^i, compute its corresponding set $C(e)$ by performing the following procedure:
   (a) Assemble the set $C(e)$ with synsets that might denote the meaning of entity e.
   (b) Create a virtual document of e, and a virtual document for every synset in $C(e)$.
   (c) Calculate the document similarities between the virtual document denoting e and the different virtual documents originating from $C(e)$.
   (d) Discard all synsets from $C(e)$ that resulted in a low similarity score with the virtual document of e, using some selection procedure.
2. Compute the WordNet similarity for all combinations of $e^1 \in E_x^1$ and $e^2 \in E_x^2$ using the processed collections $C(e^1)$ and $C(e^2)$.

The essential operation of the approach is the exclusion of synsets from the WordNet similarity calculation. This is determined using the document similarities between the virtual documents originating from the synsets and the virtual document originating from the ontology concepts. Figure 2 illustrates steps 1.b - 2 of our approach for two arbitrary ontology entities e^1 and e^2: Once the similarity matrix, meaning all pairwise similarities between the entities of both ontologies, are computed, the final alignment of the mapping process can be extracted or the matrix can be combined with other similarity matrices.

5.1 Synset Selection and Virtual Document Similarity

The initial step of the approach entails the allocation of synsets that might denote the meaning of a concept. The name of the concept, meaning the fragment of its URI, and alternate labels, when provided, are used for this purpose. While ideally one would prefer synsets which contain an exact match of the concept name or label, precautions must

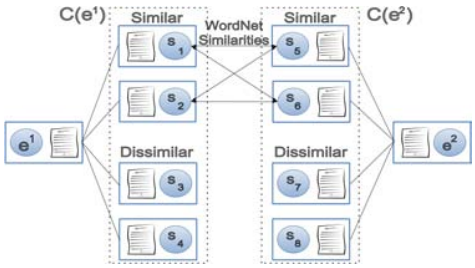


Fig. 2. Visualization of step 1.b-2 of the proposed approach for any entity e^1 from ontology O_1 and any entity e^2 from ontology 2.

be made for the eventually that no exact match can be found. For this research, several pre-processing methods have been applied such as the removal of special characters, stop-word removal and tokenization. It is possible to enhance these precautions further by for instance the application of advanced natural language techniques, however the investigation of such techniques in this context is beyond the scope of this research. When faced with ontologies that do not contain concept names using natural language, for instance by using numeric identifiers instead, and containing no labels, it is unlikely that any pre-processing technique will be able to reliably identify possible synsets, in which case a lexical similarity is ill-suited for that particular matching problem.

In the second step, the virtual document model as described in section 4 is applied to each ontology concept and to each synset that has been gathered in the previous step. The resulting virtual document are represented using the well known vector-space model [16]. In order to compute the similarities between the synset documents and the concept documents, the established cosine-similarity is applied [19].

5.2 Synset Selection

Once the similarities between the entity document and the different synset documents are known, a selection method is applied in order to only couple synsets that resulted in a high similarity value, while discarding the remaining synsets. It is possible to tackle this problem from various angles, ranging from very lenient methods, discarding only the very worst synsets, to strict methods, coupling only the highest scoring synsets. Several selection methods have been investigated for this research, such that both strict and lenient methods are tested. To test lenient selection methods, two methods using the arithmetic (A-MEAN) and geometric mean (G-MEAN) as a threshold have been investigated. Two other methods have been tested in order to investigate whether a more strict approach is more suitable. The first method, annotated as M-STD, consists of subtracting the standard deviation of the similarities from the maximum obtained similarity, and using the resulting value as a threshold. This method has the interesting property that it is more strict when there is a subset of documents that is significantly more similar than the remaining documents, and more lenient when it not as easy to identify the correct correspondences. The second investigated strict method (MAX) consists of only coupling the synset where its corresponding virtual document resulted in the highest similarity value.

5.3 WordNet Distance

After selecting the most appropriate synsets using the document similarities, the similarity between two entities can now be computed using their assigned synsets. This presents the problem of determining the similarity between two sets of synsets, where one can assume that within each of these sets resides one synset that represents the true meaning of its corresponding entity. Thus, if one were to compare two sets of synsets, assuming that the originating entities are semantically related, then one can assume that the resulting similarity between the two synsets that both represent the true meaning of their corresponding entities, should be a high value. Inspecting all pairwise similarities between all combinations of synsets between both sets should yield at least one high

similarity value. When comparing two sets originating from semantically unrelated entities, one can assume that there should be no pairwise similarity of high value present. A reasonable way of computing the similarity of two sets of synsets is to compute the maximum similarity over all pairwise combination between the two sets.

There exist several ways to compute the semantic similarity within WordNet [3] that can be applied, however finding the optimal measure is beyond the scope of this paper. Here, a similarity measure with similar properties as the Leacock-Chodorow similarity [3] has been applied. The similarity $sim(s_1, s_2)$ of two synsets s_1 and s_2 is computed using the distance function $dist(s_1, s_2)$, which determines the distance of two synsets inside the taxonomy, and the over depth D of the taxonomy:

$$sim(s_1, s_2) = \begin{cases} \frac{D - dist(s_1, s_2)}{D} & \text{if } dist(s_1, s_2) \leq D \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

This measure is similar to the Leacock-Chodorow similarity in that it relates the taxonomic distance of two synsets to the depth of the taxonomy. In order to ensure that the resulting similarity values fall within the interval of $[0, 1]$ and thus can be integrated into larger mapping systems, the log-scaling has been omitted in favor of a linear scale.

6 Experiments

In this section, the experiments that have been performed to test the effectiveness of our approach will be presented. Subsection 6.1 details an evaluation on the conference data set, originating from the Alignment Evaluation Initiative 2010 (OAEI 2010) competition [5], which demonstrates to what extent our Synset coupling method can improve a framework using WordNet. Subsections 6.2 and 6.3 compares our matcher, referred to as MaasMatch (MM), against existing frameworks using the results from the OAEI 2011 campaign [6] in which MaasMatch participated. For this research, the weighing parameters for the virtual documents were all given the equal value of 1 such that the vectors resemble document vectors originating from human-written documents, since a sensitivity analysis of these parameters is beyond the scope of this article and will be addressed in future research. The WordNet similarity matrix is combined with the similarity matrix stemming from the Jaro [8] string similarity using the average similarity of each pairwise combination, upon which the Naive descending extraction algorithm [12] is applied to generate a temporary mapping. For the experiment in subsection 6.1 a threshold of 0.7 is used, where for the OAEI 2011 evaluation a threshold of 0.95 has been applied. MaasMatch can be downloaded from the SEALS-platform, which can be accessed at `http://www.seals-project.eu/`.

When evaluating the performance of an ontology mapping procedure, the most common practise is to compare a generated alignment with a reference alignment of the same data set. Measures such as precision and recall [7], can then be computed to express the correctness and completeness of the computed alignment. Given a generated alignment A and reference alignment R, the precision $P(A, R)$ and recall $R(A, R)$ of the generated alignment A are defined as:

$$P(A, R) = \frac{R \cap A}{A} \tag{7}$$

$$R(A, R) = \frac{R \cap A}{R} \tag{8}$$

Given the precision and recall of an alignment, a common measure to express the overall quality of the alignment is the F-measure [7]. Given a generated alignment A and a reference alignment R, the F-measure can be computed al follows:

$$\text{F-measure} = \frac{2 * P(A,R) * R(A,R)}{P(A,R) + R(A,R)} \tag{9}$$

The F-measure is the harmonic mean between precision and recall. Given that these measurements require a reference alignment, they are often inconvenient for large-scale evaluations, since reference alignments require an exceeding amount of effort to create. The used data sets, however, do feature reference alignments, such that the performance of a mapping approach can easily be computed and compared.

6.1 Synset Coupling

To investigate to what extent our approach improves a framework using a WordNet similarity, we evaluated our framework using different variations of our approach on the conference data set of the OAEI 2010 competition. This data set consists of real-world ontologies describing the conference domain and contains a reference alignment for each possible combination of ontologies from this data set. Figure 3 displays the results of our approach on the conference data set. Each entry in Figure 3 denotes a different synset selection procedure, which are arranged according to their strictness, such that the most lenient method is located on the far left side and the most strict method is located on the far right. Note that the most lenient method, denoted as 'none', does not discard any synsets based on their document similarities, resulting in the equivalent of a conventional WordNet similarity, which can be used as a basis for comparison. From Figure 3 we can see two notable trends. First and foremost is the observation that the more strict the synset selection procedure is, the higher the overall performance of the matcher is, as indicated by the F-Measure. This is solely due to a steady increase of the precision of the alignments. Secondly, it is notable that the recall of the alignments decreases slightly upon increasing the strictness of the selection procedure. This can be explained by the possibility that during the selection synsets are discarded that better denote the meaning of a given concept than its similarity value indicates.

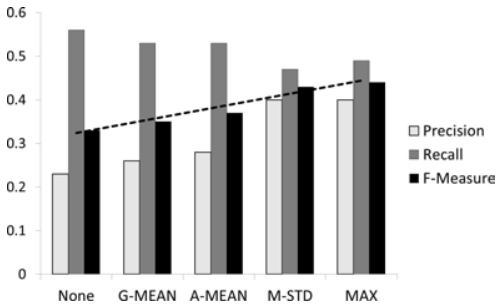


Fig. 3. Evaluation of coupling methods on the OAEI 2010 Conference data set.

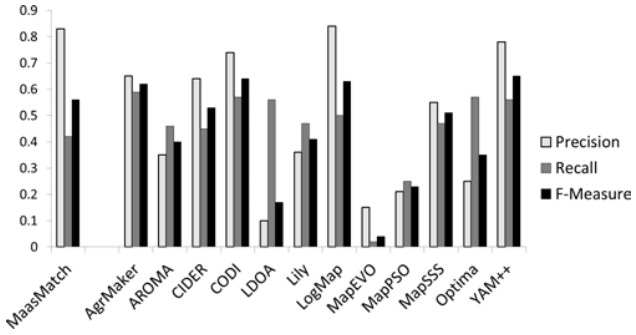


Fig. 4. Results of MaasMatch in the OAEI 2011 competition on the conference data set, compared against the results of the other participants

Overall, the highest performing variation of our coupling technique achieved an f-measure 0.44, which is an increase of 0.11 when compared to our framework without a selective coupling method. These results indicate that our coupling technique improves the computed WordNet similarities to such an extent that the computed alignments exhibit a significant increase in quality, mostly with regard to their precision.

6.2 OAEI 2011: Conference Dataset

Using the best performing synset selection method, as determined in subsection 6.1, our framework has been evaluated in the OAEI 2011 competition. The results of the evaluation on the conference data set can be seen in Figure 4. From Figure 4 one can see that MaasMatch achieved a high precision and moderate recall over the conference data set, resulting in the fifth-highest f-measure among the participants, which is above average. A noteworthy aspect of this result is that this result has been achieved by only applying lexical similarities, which are better suited at resolving naming conflicts as opposed to other conflicts. This in turn also explains the moderate recall value, since it would require a larger, and more importantly a more varied set of similarity values, to deal with the remaining types of heterogeneities as well. Hence, it is encouraging to see these good results when taking into account the moderate complexity of the framework.

6.3 OAEI 2011: Benchmark Dataset

The benchmark data set is a synthetic data set, where a reference ontology is matched with many systematic variations of itself. These variations include many aspects, such as introducing errors or randomizing names, omitting certain types of information or altering the structure of the ontology. Since a base ontology is compared to variations of itself, this data set does not contain a large quantity of naming conflicts, which our approach is targeted at. However, it is interesting to see how our framework performs when faced with every kind of heterogeneity. Figure 5 displays the results of the OAEI 2011 evaluation on the benchmark data set.

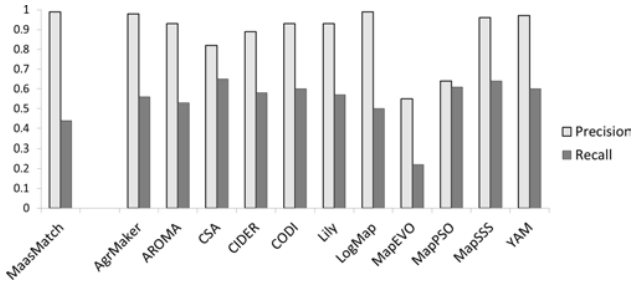


Fig. 5. Results of MaasMatch in the OAEI 2011 competition on the benchmark data set, compared against the results of the other participants

From Figure 5 we can see that the overall performance MaasMatch resulted in a high precision score and relatively low recall score when compared to the competitors. The low recall score can be explained by the fact that the WordNet similarity of our approach relies on collecting synsets using information stored in the names of the ontology concepts. The data set regularly contains ontologies with altered or scrambled names, making it extremely difficult to couple synsets that might denote the meaning of an entity. These alterations also have a negative impact on the quality of the constructed virtual documents, especially if names or annotations are scrambled or completely left out, resulting in MaasMatch performing poorly in benchmarks that contain such alterations. Despite these drawbacks, it was possible to achieve results similar to established matchers that address all types of heterogeneities. Given these results, the performance can be improved if measures are added which tackle other types of heterogeneities, especially if such measures increase the recall without impacting the precision.

7 Conclusion

In this paper, we proposed a method to improve the coupling of ontology concepts with their corresponding WordNet entries. The experiment on the OAEI 2010 data set shows that our approach increases the quality of the computed alignments, mainly with regards to their precision. Furthermore, it is established that strict coupling methods produce better results than lenient coupling methods. The result of the OAEI 2011 evaluation show that a framework using the proposed technique can compete with established frameworks, especially with regards to the conference data set. However, the results of the benchmark data set indicate a reliance on the presence of adequate concept names and descriptions. Future research can be performed on improving the robustness of our approach when given distorted names and descriptions.

The recall of the alignments slightly decreases if our approach is applied, indicating that occasionally the correct meaning of an entity is not established. A possible solution would be the improvement of the representative strength of the virtual documents. This can be achieved by refining the current virtual document model, such that for instance descriptions from different OWL types of relations receive different weights.

References

[1] S. Banerjee and T. Pedersen. Extended gloss overlaps as a measure of semantic relatedness. In *Proceedings of the 18th international joint conference on Artificial intelligence*, IJCAI'03, pages 805–810, San Francisco, CA, USA, 2003.

[2] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 284(5):34–43, 2001.

[3] A. Budanitsky and G. Hirst. Semantic distance in wordnet: An experimental, application-oriented evaluation of five measures. In *Workshop on WordNet and other lexical resources, second meeting of the NAACL*, 2001.

[4] P. Buitelaar, P. Cimianop, P. Haase, and M. Sintek. Towards linguistically grounded ontologies. In *The Semantic Web: Research and Applications*, volume 5554 of *Lecture Notes in Computer Science*, pages 111–125. Springer Berlin / Heidelberg, 2009.

[5] J. Euzenat, A. Ferrara, C. Meilicke, J. Pane, F. Scharffe, P. Shvaiko, H. Stuckenschmidt, O. Svab-Zamazal, V. Svatek, and C. Trojahn. First results of the ontology alignment evaluation initiative 2010. In *Proceedings of ISWC Workshop on OM*, 2010.

[6] J. Euzenat, A. Ferrara, R.W. van Hague, L. Hollink, C. Meilicke, A. Nikolov, F. Scharffe, P. Shvaiko, H. Stuckenschmidt, O. Svab-Zamazal, and C. Trojahn dos Santos. Results of the ontology alignment evaluation initiative 2011. In *Proc. 6th ISWC workshop on ontology matching (OM), Bonn (DE)*.

[7] F. Giunchiglia, M. Yatskevich, P. Avesani, and P. Shvaiko. A large dataset for the evaluation of ontology matching. *Knowl. Eng. Rev.*, 24:137–157, June 2009.

[8] M. Jaro. Advances in record-linkage methodology as applied to matching the 1985 census of tampa, florida. *J. of the American Statistical Association*, 84(406):pp. 414–420, 1989.

[9] O. Lassila, R. R. Swick, and W3C. Resource description framework (rdf) model and syntax specification, 1998.

[10] M. Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th annual international conference on Systems documentation*, SIGDOC '86, pages 24–26, 1986.

[11] D. L. McGuinness and F. van Harmelen. OWL web ontology language overview. W3C recommendation, W3C, February 2004.

[12] C. Meilicke and H. Stuckenschmidt. Analyzing mapping extraction approaches. *The Second International Workshop on Ontology Matching*, 2007.

[13] G. A. Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38:39–41, November 1995.

[14] R. Navigli. Word sense disambiguation: A survey. *ACM Comput. Surv.*, 41(2):10:1–10:69, February 2009.

[15] Y. Qu, W. Hu, and G. Cheng. Constructing virtual documents for ontology matching. In *Proceedings of the 15th international conference on World Wide Web*, WWW '06, pages 23–31, New York, NY, USA, 2006. ACM.

[16] G. Salton, A. Wong, and C.S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18:613–620, November 1975.

[17] F. C. Schadd and N. Roos. Improving ontology matchers utilizing linguistic ontologies: an information retrieval approach. In *Proceedings of the BNAIC 2011*, 2011.

[18] P. Shvaiko and J. Euzenat. A survey of schema-based matching approaches. In *Journal on Data Semantics IV*, volume 3730, pages 146–171. 2005.

[19] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison Wesley, 1 edition, May 2005.

[20] C. Watters. Information retrieval and the virtual document. *J. Am. Soc. Inf. Sci.*, 50:1028–1029, September 1999.

WikiMatch – Using Wikipedia for Ontology Matching

Sven Hertling, Heiko Paulheim

Technische Universität Darmstadt
Knowledge Engineering Group
hertling@ke.tu-darmstadt.de, paulheim@ke.tu-darmstadt.de

Abstract. Finding correspondences between different ontologies is a crucial task in the Semantic Web. Ontology matching tools are capable of solving that task in an automated manner, some even dealing with ontologies in different natural languages. Most state of the art matching tools use internal element and structure based techniques, while the use of large-scale external knowledge resources, especially internet resources, is still rare. In this paper, we introduce *WikiMatch*, a matching tool that exploits Wikipedia as an external knowledge source. We show that using Wikipedia is a feasible way of performing ontology matching, especially if different natural languages are involved.

Keywords: Ontology Matching, Ontology Alignment, Wikipedia, External Resources

1 Introduction

Ontologies are an essential building block of the Semantic Web. They formally describe the vocabulary used in a domain in a machine-interpretable way. Thus, ontologies can be used to unambiguously exchange information between machines. If multiple ontologies are used in parallel in a domain, e.g., when integrating two data sets, mappings between those ontologies are required.

Approaches for finding such mappings or alignments automatically are called *ontology matching* approaches [8]. Possible applications the integration of different data sets, the discovery of heterogeneously described web services, or the exchange of business data between business partners.

Many state of the art matchers are based on *internal* techniques, i.e., they only use the knowledge contained in the ontologies to match, but no external knowledge sources. Such matchers compare local names and labels of elements contained in the ontologies, and use structural features. In contrast, *external* techniques make use of external resources, such as synonym lists, dictionaries, or linguistic resources such as wordnet. In many cases, e.g., for recognizing synonyms, they produce useful results.

On the other hand, such matchers are restricted to the domain of the resources they use, and most of the resources are limited and outdated if not maintained by the tool developer.

In this paper, we present *WikiMatch*, an ontology matching approach based on Wikipedia as an external resource. The knowledge in Wikipedia is based on 23 million articles written by volunteers around the world, covering almost every possible domain at least to a certain depth. Wikipedia pages exist in 285 languages with links between articles in different languages[1], hence, it is also usable for matching ontologies in different languages.

The goal of our approach is to make this large knowledge source usable for ontology matching. To that end, we present an approach that exploits Wikipedia's search functionality and inter-language links for finding mappings between ontologies.

The rest of this paper is structured as follows. Section 2 introduces the state of the art and discussed related matching approaches. Section 3 describes our approach and two different variants, and section 4 shows the evaluation results for both variants, using ontologies and reference alignments from the Ontology Alignment Evaluation Initiative (OAEI)[2]. This paper is then summarized by a conclusion and an outlook on future work.

2 Related Work

Discovering and using relevant sources of background knowledge has been named as one of the ten main challenges to ontology matching [22]. One possible approach is the use of *upper ontologies*, i.e., general purpose ontologies, as background knowledge. Such approaches try to find mappings between two ontologies by relating their elements to a comprehensive upper level ontology and then computing similarities within that ontology between the mapped terms [15]. As such, upper ontologies may be detailed and contain rich information, e.g., about alternative names or spellings for concepts, this approach can help increasing the result quality of ontology matching.

Most approaches employing upper ontologies only use a small set of fixed comprehensive ontologies, such as *Proton* in *BLOOMS+* [11], or *SUMO* in *LOM* [13]. Some authors have also discussed the potential of using domain-specific upper level ontologies for matching tasks in certain domains [1]. In contrast, Sabou et al. [21] have discussed a generic approach using dynamic discovery of suitable external ontologies by employing the ontology search engine Swoogle[3]. This search engine is employed to find suitable ontologies to be used as upper ontologies in the matching process.

Apart from upper-level ontologies, another widely used external knowledge source are linguistic resources such as thesauri, e.g. *WordNet* [17]. Those resources contain synonym definitions, typical relations between words, or multilingual translations. State of the art tools using such resources comprise, e.g., *AgreementMaker* [7], *LogMap* [12], and *YAM++* [19]. In [5], the use of domain-specific, semi-structured corpora of documents is discussed as a means to ontol-

[1] `http://stats.wikimedia.org`
[2] `http://oaei.ontologymatching.org/`
[3] `http://swoogle.umbc.edu/`

ogy matching in specific domains, but it requires the availability and pre-selection of such documents.

Web data is rarely used in ontology matching. One of the few approaches is *COMS*, which uses the online source *Wiktionary*[4] as lexical background knowledge, and employs the Google Translation API[5] for addressing multi-lingual ontologies [14]. The use of the Google Translation API for multi-lingual ontologies has also been proposed by Fu et al. [9] and Trojahn et al. [23]. Furthermore, the use of Google for synonym detection has been announced for *MapSSS* [4], but not implemented and evaluated to date.

Gligorov et al. have discussed the use of the Google search engine for ontology matching [10]. They use the *Google similarity distance* [6] to compare the similarity of two terms is computed from the number of search results for each of the terms alone, and the terms in combination. Since that approach requires a quadratic number of search engine calls, it does not scale well to larger problems.

Wikipedia, despite being one of the largest cross-domain knowledge collections, and also one of the best-known, has been rarely explored as a source of background knowledge in ontology matching so far. *BLOOMS* uses only Wikipedia's category tree and employs it as an upper ontology (see above), rather than exploiting Wikipedia as a whole. In [3], the exploitation of Wikipedia's cross-language links has been discussed as a means for addressing cross-language ontology matching. In [2], the use of Wikipedia as a large-scale text corpus for ontology alignment has been proposed, but no implemented prototype and evaluation are provided.

3 Approach

The basic idea of our approach is to use Wikipedia's search engine to retrieve a result set of Wikipedia articles describing the term. To support multilingual scenarios, we retrieve all language links per article in a second translation step. Since the article titles are unique for every Wikipedia in one language, we compare the sets of retrieved titles to compute the similarity between two concepts.

Wikipedia is based on a software platform called MediaWiki, written in PHP. This framework is used to run Wikipedia, Wiktionary, Wikinews, and so on. MediaWiki offers an API[6] for all pages which run on this software. That API offers two different search engines which vary in their purpose. The traditional search engine performs a full text search, while OpenSearch is used to assist users with suggestions when typing their search.

The standard search engine[7] performs a full text search in all articles. If a term is misspelled, the response contains a suggestion with for the correctly spelled word. The goal of the search engine is to find Wikipedia articles that contain all words from the input.

[4] `http://www.wiktionary.org/`
[5] `http://code.google.com/apis/language/translate/overview.html`
[6] `http://www.mediawiki.org/wiki/API:Main_page`
[7] `http://www.mediawiki.org/wiki/API:Search`

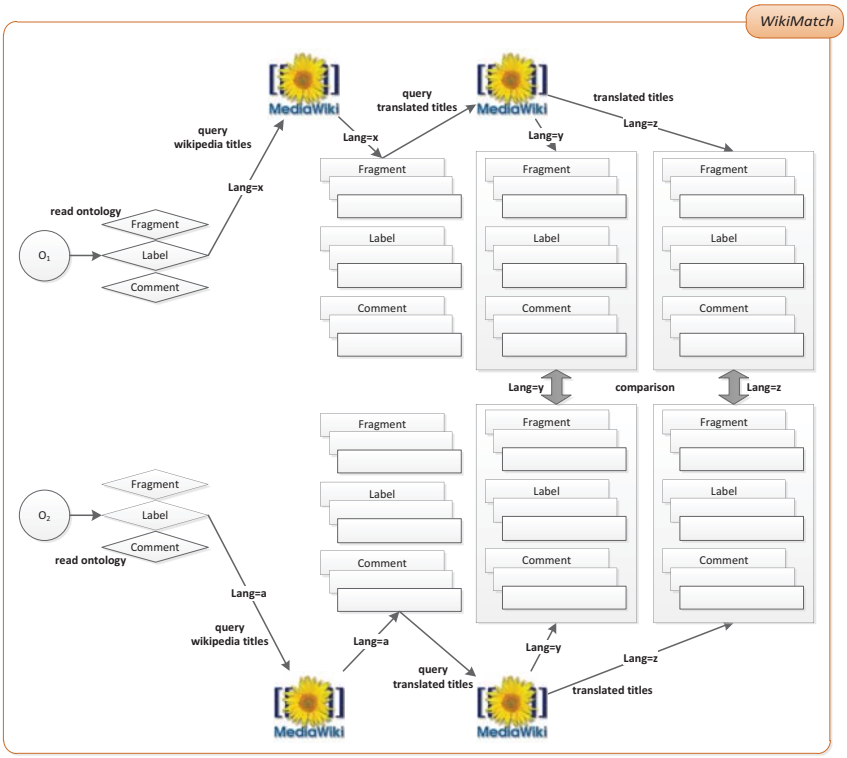


Fig. 1. Illustration of the matching process. For every fragment, label and comment we query wikipedia titles and retrieve all language links as a second step. We compare the requested titles with the same language and returns the maximum of the cross product from fragment, label and comment.

The other search engine is *Opensearch*[8]. It is used for suggesting some terms while the user is typing in the search box on Wikipedia. This search is not applicable for our task, because we already have the full term and hence do not need any hint on possible completions. In our preliminary experiments, that search engine did not work well. Often, labels and comments are composed of many words (tokens), and in that case, the reply of Opensearch is empty[9]. Thus, we use Wikipedia's standard search engine.

For performing searches, we use the URI fragments, labels, and comments of each concept as input strings to the search input. Since the search engine tries to

[8] `http://www.mediawiki.org/wiki/API:Opensearch`

[9] See this example taken from the OAEI conference track: `http://en.wikipedia.org/w/api.php?action=opensearch&search=Subject%20Area&limit=10&namespace=0&format=jsonfm`

find *all* the words in the input query in a Wikipedia article and does not ignore stop words, we remove stop words in a preprocessing step. After preprocessing, our approach uses two different variants for performing the search: taking the preprocessed strings as search input, or searching for each individual token in the input string.

For example, from the label *member of the program committee*, the stop words *of* and *the* would be removed in a first step. Our standard search approach then searches for *member program committee*, while the individual token search approach would trigger three searches for *member*, *program*, and *committee*.

In both search variants, we divide the ontology elements to match into three sets: classes, datatype properties, and object properties. We match only concepts of the same part. This yields on the one hand in better performance and higher precision and on the other hand all mappings are consistent with OWL Lite/DL. Additionally, we can adjust individual threshold values for every type of mapping, like class-class or property-property mappings.

For every ontology concept we extract the fragment, labels and comments, and compare each combination of concepts from both ontologies. Each fragment, label, and comment (or in the individual token search approach, each token thereof) is sought via the Wikipedia search engine. The result is a set of documents per fragment, label, and comment. From those sets, the similarity of two concepts is computed. We compare only titles with the same language. Given that the search for a term t – which can be a label, URI fragment, or comment – returns a set $S(t)$ of Wikipedia article titles (which can be in any language), the similarity between two concepts (i.e., classes and properties) c_1 and c_2 from two ontologies O_1 and O_2 is defined as

$$max_{t_i \in \{label(c_i), fragment(c_i), comment(c_i)\}, i \in \{1,2\}} \frac{\#(S(t_1) \cap S(t_2))}{\#(S(t_1) \cup S(t_2))} \quad (1)$$

If the similarity exceeds a certain threshold, we return a mapping element for the two concepts.

The sets of Wikipedia articles are retrieved by first searching for the Wikipedia article, and then retrieving all translations to other articles in a second step. Thus, our approach treats both single-language and multi-language ontology matching problems the same.

To address the correct search engine, we extract the ontologies' language tags and create a URL like *http://(lang-tag).wikipedia.org/w/api.php*. To this URL we send a request for n titles[10]. The results are all in the language we extracted from the ontology. To compare titles from other languages, we add all *language-links* appear on the requested wikipedia pages. If the answer contains a suggestion for a spelling correction, we make another query in order to get better result for misspelled words. Figure 1 depicts the matching process in a schematic way.

As the simple search approach uses the Wikimedia search interface in a trivial way, requesting articles for whole strings such as *Member of the Program*

[10] For the evaluations in this paper, we have set $n = 50$

```
float getsimilarity(term1, term2) {
  titlesForTerm1 = getAllTitles(term1);
  titlesForTerm2 = getAllTitles(term2);

  commonTitles = intersectionOf(titlesForTerm1, titlesForTerm2);
  allTitles = unionOf(titlesForTerm1, titlesForTerm2);

  return #(commonTitles) / #(allTitles);
}

List<WikipediaPage> getAllTitles(searchTerm) {
  removeStopwords(searchTerm);
  removePunctuation(searchTerm);

  if(simpleSearch) {
    resultList = searchWikipedia(searchTerm);
  }

  if(individualTokenSearch) {
    tokens = tokenize(searchTerm);
    for each token in tokens
      resultList = resultList + searchWikipedia(searchTerm);
  }

  for each page in results
    resultList = resultList + getLanguageLinks(page);

  return resultList;
}
```

Fig. 2. Algorithm for exploiting Wikipedia in Ontology Matching

Committee. Since especially comments can be fairly long, we have also implemented an alternative variant searching for *individual tokens* in the names, such as *member*, *program*, *committee*. This approach is expected to increase the recall, but maybe yield lower precision. Figure 2 shows the algorithm for both search approaches in pseudo code.

4 Evaluation

We have evaluated our tool with benchmarks from the OAEI matching campaign[11]. In this paper, we will focus on the real-world use case from the conference domain (conference), as well as the multi-lingual dataset (multifarm).

[11] `http://oaei.ontologymatching.org/`

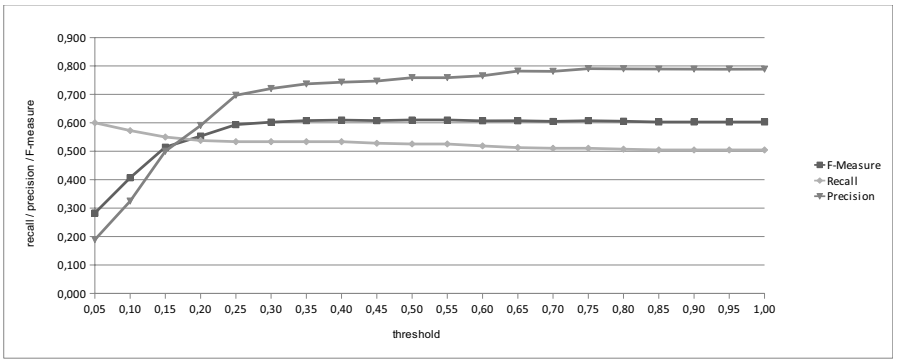


Fig. 3. Average F-measure, precision and recall for OEAI conference track with *Simple Search Approach*.

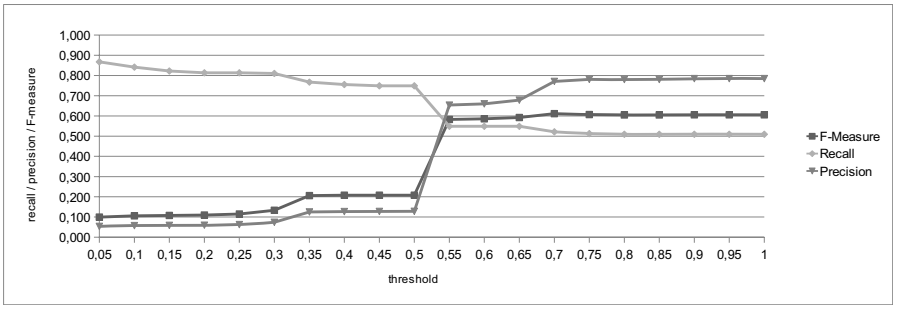


Fig. 4. Average F-measure, precision and recall for OEAI conference track with *Individual Token Search Approach*.

4.1 Evaluation on Conference Track

The conference track consists of 16 ontologies about the domain of conferences. Each of those ontologies are compared in a pairwise setting. For self-evaluation, a subset of seven ontologies is given with reference alignments, thus resulting in 21 possible test cases combinations. The following results are based on those 21 test cases and show average values over all 21 cases.

All the following diagrams are organized as follows: On the x-axis, we use different values for the threshold t, and depict recall, precision, and f-measure for those different values.

Figure 3 shows the results for the *Simple Search Approach*. The maximum f-measure of 0.610 is reached when using a threshold of 0.5. In general, for threshold values above 0.25, there are only small variations in f-measure.

Figure 4 shows the results achieved with *Individual Token Search*. There is a significant leap between 0.5 and 0.55. With a threshold of 0.5 we only get a f-measure of 0.208, while ith a threshold of 0.55, the f-measure rises to 0.582. The maximum f-measure that can be reached with this approach is 0.611, using a threshold of 0.7.

These results show that for the conference track, both approaches converge to about the same maximum f-measure when setting an appropriate threshold. If we compare our result to the OAEI 2011.5 results[12], *WikiMatch* is on the fourth rank, between *CODI* and *Hertuda*, and in particular performs significantly better than the baseline comparing concepts based on string similarity and stop word filtering.

Our matching time for the *Simple Search Approach* is 1340 seconds, which is 22 minutes and 20 seconds. *Individual Token Search Approach* takes a little bit longer. It was about 1454 seconds (24 minutes and 14 seconds).

4.2 Evaluation on Multifarm Track

The multifarm dataset is designed for multilingual ontology matching. It is based on the conference dataset described above, which is translated into eight different languages, i.e., Chinese, Czech, Dutch, French, German, Portuguese, Russian, and Spanish [16].

The evaluation of *Simple Search Approach* is depicted in figure 5. It shows that the maximum f-measure of 0.210 is reached for a threshold of 0.06, with recall decreasing at a great pace. The maximum recall is only 0.35.

Figure 6 shows the results that can be achieved on the multifarm track with individual token search. The maximum f-measure that can be achieved is 0.179 at a threshold of 0.06, with recall and precision behaving similarly to simple search. Thus, for multi-lingual problems, simple search yields slightly better results. The results are competitive with the top 5 tools at the recent OAEI 2011.5 evaluation[13].

Table 1 depicts results on the multifarm track, showing those language pairs that were part of the OAEI evaluation 2011.5 (i.e., excluding Chinese and Russian).

4.3 Performance Evaluation and Scalability

Requesting web resources at run-time usually generates run times that are not competitive internal matching approaches or external matching approaches using only local resources. On the conference and multifarm datasets, a pair of ontologies takes about 360 seconds with simple search and 450 seconds[14] with individual token search to process.

However, in contrast to approaches using co-occurence analysis on Wikipedia [18] or Google Distance [6], which require a quadratic number of search requests, our approach only issues a linear number of search requests, since it only searches for concepts in the ontologies, not for combinations of such concepts. Thus, despite being slower than other approaches, it is scalable to larger matching problems.

[12] `http://oaei.ontologymatching.org/2011.5/results/conference/index.html`
[13] `http://oaei.ontologymatching.org/2011.5/results/multifarm/index.html`
[14] On a Windows 7 64bit PC with an Intel i7(3.4 GHz) processor and 8 GB RAM

	Different ontologies (type i)			Same ontologies (type ii)		
language-pair	F-measure	precision	recall	F-measure	precision	recall
cz-de	0.250	0.295	0.247	0.140	0.488	0.083
cz-en	0.245	0.289	0.233	0.179	0.495	0.110
cz-es	0.269	0.292	0.269	0.147	0.452	0.089
cz-fr	0.211	0.264	0.191	0.143	0.463	0.085
cz-nl	0.219	0.259	0.208	0.152	0.508	0.091
cz-pt	0.157	0.189	0.163	0.106	0.308	0.066
de-en	0.290	0.280	0.345	0.252	0.475	0.173
de-es	0.256	0.259	0.301	0.198	0.423	0.134
de-fr	0.275	0.278	0.307	0.200	0.516	0.126
de-nl	0.277	0.310	0.283	0.224	0.587	0.141
de-pt	0.230	0.218	0.276	0.154	0.345	0.100
en-es	0.281	0.265	0.350	0.279	0.489	0.198
en-fr	0.283	0.290	0.315	0.257	0.550	0.171
en-nl	0.304	0.303	0.344	0.237	0.526	0.155
en-pt	0.263	0.250	0.340	0.257	0.431	0.185
es-fr	0.248	0.217	0.312	0.260	0.485	0.179
es-nl	0.224	0.224	0.242	0.224	0.516	0.143
es-pt	0.272	0.207	0.472	0.299	0.453	0.231
fr-nl	0.282	0.252	0.348	0.233	0.529	0.150
fr-pt	0.203	0.159	0.311	0.228	0.382	0.164
nl-pt	0.185	0.163	0.254	0.173	0.315	0.120
average	0.249	0.251	0.291	0.207	0.464	0.138

Table 1. Evaluation on Multifarm track with language and type specific results (*Simple Search Approach*). The threshold is 0.06. The bottom line shows the average of all language pairs.

4.4 Further Observations

There are certain cases that WikiMatch is capable of covering well, while there are others which are more problematic. In the single language scenario, as expected, class names that are equal or similar are matched without problems. For example *sponsorship* vs. *sponzorship* is matched based on the Wikipedia search engine suggesting an alternate spelling. Complex terms such as *member of the program committee* and *program committee member* can also be matched, since stopwords such as *of* and *the* are removed first, as described above. On the other hand, complex property names, such as *has written* and *is author of* are problematic for our approach, since the result lists for *written* and *author* are very dissimilar.

In the multi-lingual case, simple translations such as *Stadt(de)* and *city(en)* and close translations such as *Bankett(de)* and *dinner banquet(en)* can be handled by WikiMatch, as well as property names such as *hat E-Mailadresse(de)* and *has email(en)*. Cases where the translated terms are different, e.g., *Autor von(de)* and *has written(en)*, are equally problematic as in the single language case.

For the multi-lingual case, we have further analyzed the relation between the different language Wikipedia's sizes and the F-Measure achieved. F-Measure and recall are strongly correlated with the Wikipedia's sizes; the best results are

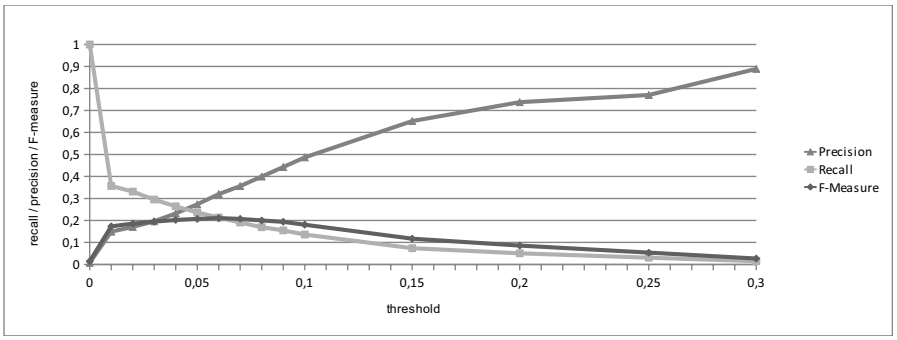


Fig. 5. Average F-measure, precision and recall for OEAI multifarm track with *Simple Search Approach*.

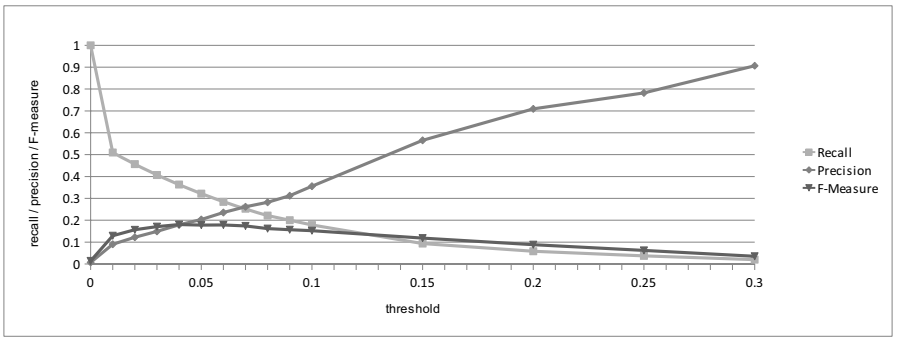


Fig. 6. Average F-measure, precision and recall for OEAI multifarm track with *Individual Token Search Approach*.

achieved with the biggest Wikipedias (English, French, German and Dutch are larger than 1,000,000 entries, and the results between those four languages are the best ones). Conversely, the worst result is achieved for Czech and Portuguese, where the Czech and the Portuguese Wikipedia are the smallest ones among the languages used in multifarm.

5 Conclusion and Future Work

In this paper, we have presented *WikiMatch* as a matching approach based on a large external resource, namely Wikipedia. We use this information to handle synonyms and determine a score describing the equality of two concepts. This paper showed that a matcher using only one external resource without respect to structural information within the ontologies can also yield good results in OAEI benchmarks.

We can handle all domains which Wikipedia covers. Since Wikipedia is community project, it is maintained by many volunteers around the world and grows to more domains each day. Thus, we do not have to take care about updates of the external resources used in in our matcher.

Moreover, we use language links to match ontologies in different languages. Since these links are maintained by humans and not created by bots, we can heavily rely on these links for translation. In this case it is also possible to cope with matching multilingual ontologies.

The aim of this paper was to explore the possibilities of exploiting Wikipedia as an external resource for ontology matching. Since the results are promising and the approach is capable of tackling many hard-to-handle cases, especially in the multi-lingual area, combining the *WikiMatch* approach with other techniques, such as structure-based matching algorithms, would be a natural step to further exploit the approach's potential.

In our experiments, we have compared two different variants for searching contents in Wikipedia, which lead to similar maximum f-measure values, but behave differently in detail. A suitable combination of both approaches could help generating better overall results.

At the moment, despite using various caches, our approach is not very fast. The most time consuming operation is querying Wikipedia. On the other hand, our approach is purely element based, which allows for efficient distribution of the matching problem to many computers [20]. Developing a parallel version of *WikiMatch* would thus eliminate that problem.

While Wikipedia is for sure one of the largest and encompassing online resource, implementing our approach with other such resources, such as *answers.com*, or exploiting even general web search engines, would be an interesting experiment to further assess the value of Wikipedia as a knowledge resource in ontology matching.

In summary, we have shown that a simple approach with Wikipedia as an external resource can handle many different problems in the ontology matching area. Especially, the matching of multilingual ontologies are covered with this approach. The external resource is never outdated and can be used for all domains covered by Wikipedia. We hope that our work will improve future ontology matcher to get better results in monolingual as well as multilingual matching.

References

1. Aleksovski, Z., ten Kate, W., van Harmelen, F.: Ontology matching using comprehensive ontology as background knowledge. In: 1st International Workshop on Ontology Matching (OM 2006). (2006) 13–24
2. Beisswanger, E.: Exploiting Relation Extraction for Ontology Alignment. In Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., 0007, L.Z., Pan, J.Z., Horrocks, I., Glimm, B., eds.: International Semantic Web Conference (2). Volume 6497 of Lecture Notes in Computer Science., Springer (2010) 289–296
3. Bouma, G.: Cross-lingual Ontology Alignment using EuroWordNet and Wikipedia. In: Proceedings of the 7th Conference on International Language Resources and Evaluation (LREC 2010). (2010) 1023–1028

4. Cheatham, M.: MapSSS Results for OAEI 2011. In: Sixth International Workshop on Ontology Matching (OM 2011). (2011)
5. Cheng, C., Lau, G., Pan, J., Law, K., Jones, A.: Domain-specific ontology mapping by corpus-based semantic similarity. In: NSF CMMI Engineering Research and Innovation Conference. (2008)
6. Cilibrasi, R., Vitanyi, P.: The google similarity distance. Knowledge and Data Engineering, IEEE Transactions on **19**(3) (2007) 370–383
7. Cruz, I.F., Antonelli, F.P., Stroe, C.: AgreementMaker: Efficient Matching for Large Real-World Schemas and Ontologies. PVLDB **2**(2) (2009) 1586–1589
8. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer, Berlin, Heidelberg, New York (2007)
9. Fu, B., Brennan, R., O'Sullivan, D.: Cross-Lingual Ontology Mapping – An Investigation of the Impact of Machine Translation. In: Fourth Asian Conference (ASWC 2009). (2009) 1–15
10. Gligorov, R., Aleksovski, Z., ten Kate, W., van Harmelen, F.: Using Google Distance to weight approximate ontology matches. In: 16th International World Wide Web Conference (WWW2007). (2007)
11. Jain, P., Yeh, P., Verma, K., Vasquez, R., Damova, M., Hitzler, P., Sheth, A.: Contextual ontology alignment of lod with an upper ontology: A case study with proton. In: Extended Semantic Web Conference (ESWC 2011), Springer (2011) 80–92
12. Jiménez-Ruiz, E., Grau, B.C.: LogMap: Logic-based and Scalable Ontology Matching. In: 10th International Semantic Web Conference (ISWC 2011). (2011) 273–288
13. Li, J.: LOM: A Lexicon-based Ontology Mapping Tool. In: Proceedings of the Performance Metrics for Intelligent Systems (PerMIS. (2004) 2004
14. Lin, F., Krizhanovsky, A.: Multilingual Ontology Matching based on Wiktionary Data Accessible via SPARQL Endpoint. In: Russian Conference on Digital Libraries 2011 (RCDL'2011). (2011) 1–8
15. Mascardi, V., Locoro, A., Rosso, P.: Automatic Ontology Matching via Upper Ontologies: A Systematic Evaluation. IEEE Transactions on Knowledge and Data Engineering **22** (2010) 609–623
16. Meilicke, C., Castro, R.G., Freitas, F., van Hage, W.R., Montiel-Ponsoda, E., de Azevedo, R.R., Stuckenschmidt, H., Šváb-Zamazal, O., Svátek, V., Tamilin, A., Trojahn, C., Wang, S.: MultiFarm: A Benchmark for Multilingual Ontology Matching. Journal of Web Semantics (2012)
17. Miller, G.A.: WordNet: A Lexical Database for English. Communications of the ACM **38**(11) (1995) 39–41
18. Nakayama, K., Hara, T., Nishio, S.: Wikipedia mining for an association web thesaurus construction. Web Information Systems Engineering–WISE 2007 (2007) 322–334
19. Ngo, D., Bellahsene, Z., Coletta, R.: YAM++ – Results for OAEI 2011. In: Sixth International Workshop on Ontology Matching (OM 2011). (2011)
20. Paulheim, H.: On Applying Matching Tools to Large-scale Ontologies. In: Third International Workshop on Ontology Matching (OM-2008). (2008)
21. Sabou, M., d'Aquin, M., Motta, E.: Exploring the semantic web as background knowledge for ontology matching. Journal on Data Semantics **11** (2008) 156–190
22. Shvaiko, P., Euzenat, J.: Ten Challenges for Ontology Matching. In: On the Move to Meaningful Internet Systems: OTM 2008. (2008) 1164–182
23. Trojahn, C., Quaresma, P., Vieira, R.: An API for Multilingual Ontology Matching. In: Proceedings of the 7th Edition of the Language Resources and Evaluation Conference, Malta (2010) 3830–3835

RIO: Minimizing User Interaction in Debugging of Aligned Ontologies *

Patrick Rodler, Kostyantyn Shchekotykhin, Philipp Fleiss, and Gerhard Friedrich

Alpen-Adria Universität, Klagenfurt, 9020 Austria
firstname.lastname@aau.at

Abstract. Efficient ontology debugging is a cornerstone for many activities in the context of the Semantic Web, especially when automatic tools produce (parts of) ontologies such as in the field of ontology matching. The best currently known interactive debugging systems rely upon meta information in terms of fault probabilities, which can speed up the debugging procedure in the good case, but can also have negative impact in the bad case. Unfortunately, assessment of meta information is only possible a-posteriori. Hence, as long as the actual fault is unknown, there is always some risk of suboptimal interactive diagnoses discrimination. As an alternative, one might prefer to rely on a no-risk strategy. In this case, however, possibly well-chosen meta information cannot be exploited, resulting again in inefficient debugging actions. In this work we present a reinforcement learning strategy that continuously adapts its behavior depending on the performance achieved and minimizes the risk of using low-quality meta information. Therefore, this method is suitable for application scenarios where reliable a-priori fault estimates are difficult to obtain. Using faulty ontologies produced by ontology matchers, we show that the proposed strategy outperforms both active learning and no-risk approaches on average w.r.t. required amount of user interaction.

1 Introduction

The foundation for widespread adoption of Semantic Web technologies is a broad community of ontology developers which is not restricted to experienced knowledge engineers. Instead, domain experts from diverse fields should be able to create ontologies incorporating their knowledge as autonomously as possible. The resulting ontologies are required to fulfill some minimal quality criteria, usually consistency, coherency and no undesired entailments, in order to grant successful deployment. However, the correct formulation of logical descriptions in ontologies is an error-prone task which accounts for a need for assistance in ontology development in terms of ontology debugging tools. Things get even worse when independent standalone ontologies describing related domains are unified to a single ontology (called aligned ontology) by adding a set of suitable correspondences (called alignment) between signature elements of the different ontologies. This task is addressed in the field of ontology matching where researchers aim to produce automated tools for the generation of correspondences. Applying such tools, however, often results in inconsistent/incoherent aligned ontologies even though input ontologies (considered separately) do not violate any quality criteria. Moreover, these aligned ontologies may exhibit a very complex fault structure as a consequence of (1) adding many links between the single ontologies at once and since

* This research is funded by Austrian Science Fund (Project V-Know, contract 19996).

(2) the actual fault may be located in the produced alignment and/or in one or more of the single ontologies, e.g. if a correct correpondence between two concepts "activates" a fault in one of the single ontologies. In this vein, many different sources of inconsistency/incoherence could arise (due to (1)) each of which may comprise parts of each single ontology as well as of the alignment (due to (2)). In this work we present an interactive approach dealing with the general problem of locating a fault throughout the entire aligned ontology, not just in the alignment, as addressed by state-of-the-art systems in ontology matching such as CODI [10] or LogMap [5]. Comparison of our method with these systems is inappropriate since they use greedy diagnosis techniques (e.g. [8]), whereas our approach is complete.

Usually, ontology debugging tools [3, 4] use model-based diagnosis [11] to identify sets of faulty axioms, called diagnoses, that need to be modified or deleted in order to meet the imposed quality requirements. The major challenge inherent in the debugging task is often a substantial number of alternative diagnoses. This problem has been addressed in [12] by proposing an active learning debugging method which queries the user (e.g. a domain expert) for additional information about the intended ontology.

In a debugging scenario involving a faulty ontology developed by one user, the meta information might be extracted from the logs of previous sessions, if available, or specified by the user based on their experience w.r.t. own faults. However, in scenarios involving automatized systems producing (parts of) ontologies as in ontology matching, the choice of reasonable meta information is rather unclear. If, on the one hand, an active learning method is used relying on a guess of the meta information, this might result in an overhead w.r.t. user interaction of more than 2000%. If one wants to play it safe, on the other hand, by deciding not to exploit any meta information at all, this might also result in substantial extra time and effort for the user. So, in the light of current state-of-the-art one is spoilt for choice between debugging strategies with high potential but also high risk, or methods with no risk but also no potential.

In this work we present an ontology debugging approach with high potential and low risk, which allows to minimize user interaction throughout a debugging session on average, without depending on high-quality meta information. By virtue of its reinforcement learning capability, our approach is optimally suited for debugging aligned ontologies, where only vague or no meta information is available. On the one hand, our method takes advantage of the given meta information as long as good performance is achieved. On the other hand, it gradually gets more independent of meta information if suboptimal behavior is measured. The method constantly improves the quality of meta information and adapts a risk parameter based on the new information obtained by querying the user. This means that, in case of good meta information, the performance of our method will be close to the performance of the active learning method, whereas, in case of bad meta information, the achieved performance will approach the performance of the risk-free strategy. So, our approach can be seen as a risk optimization strategy (RIO) which combines the benefits of active learning and risk-free strategies. Experiments on two datasets of faulty ontologies produced by ontology matching systems show the feasibility, efficiency and scalability of RIO. The evaluation of these experiments will manifest that, on average, RIO is the best choice of strategy for both good and bad meta information with savings in terms of user interaction of up to 80%.

The problem specification, basic concepts and a motivating example are provided in Section 2. Section 3 explains the suggested approach and gives implementation details. Evaluation results are described in Section 4. Section 5 concludes.

2 Basic Concepts and Motivation

Ontology debugging deals with the following problem: Given is an ontology $\mathcal{O}$ which does not meet postulated requirements R.[1] $\mathcal{O}$ is a set of axioms formulated in some monotonic knowledge representation language, e.g. OWL. The task is to find one of generally many alternative subsets of axioms in $\mathcal{O}$, called target diagnosis $\mathcal{D}_t \in \mathcal{O}$, that needs to be altered or eliminated from the ontology such that the resulting ontology meets the given requirements and has the intended semantics. The general debugging setting we consider also envisions the opportunity for the user to specify some background knowledge $\mathcal{B}$, i.e. a set of axioms which are known to be correct. Moreover, we allow definition of a set P of positive (entailed) and a set N of negative (non-entailed) test cases, where each test case is a set of axioms.

More formally, ontology debugging can be defined in terms of conditions a target ontology must fulfill, which leads to the definition of a diagnosis problem instance, for which we search for solutions, i.e. diagnoses:

Definition 1 (Target Ontology, Diagnosis Problem Instance). *Let $\mathcal{O} = (\mathcal{T}, \mathcal{A})$ denote an ontology consisting of a set of terminological axioms $\mathcal{T}$ and a set of assertional axioms $\mathcal{A}$, P a set of positive test cases, N a set of negative test cases, $\mathcal{B}$ a set of background knowledge axioms, and R a set of requirements to an ontology. Then an ontology $\mathcal{O}_t$ is called* target ontology *iff all the following conditions are fulfilled:*

$$\forall\, r \in R:\ \mathcal{O}_t \cup \mathcal{B} \textit{ fulfills } r$$
$$\forall\, p \in P:\ \mathcal{O}_t \cup \mathcal{B} \models p$$
$$\forall\, n \in N:\ \mathcal{O}_t \cup \mathcal{B} \not\models n$$

The tuple $\langle \mathcal{O}, \mathcal{B}, P, N \rangle_R$ is called a diagnosis problem instance *iff $\mathcal{B} \cup (\bigcup_{p \in P} p) \not\models n$ for all $n \in N$ and $\mathcal{O}$ is not a target ontology, i.e. $\mathcal{O}$ violates at least one of the conditions above.*

Definition 2 (Diagnosis). *We call $\mathcal{D} \subseteq \mathcal{O}$ a* diagnosis *w.r.t. a diagnosis problem instance $\langle \mathcal{O}, \mathcal{B}, P, N \rangle_R$ iff there exists a set of axioms $EX_{\mathcal{D}}$ such that $(\mathcal{O} \setminus \mathcal{D}) \cup EX_{\mathcal{D}}$ is a target ontology. A diagnosis $\mathcal{D}$ is* minimal *iff there is no $\mathcal{D}' \subset \mathcal{D}$ such that $\mathcal{D}'$ is a diagnosis. A diagnosis $\mathcal{D}$ gives complete information about the correctness of each axiom $ax_k \in \mathcal{O}$, i.e. all $ax_i \in \mathcal{D}$ are assumed to be faulty and all $ax_j \in \mathcal{O} \setminus \mathcal{D}$ are assumed to be correct. The set of all minimal diagnoses is denoted by* $\mathbf{D}$.

The identification of an extension $EX_{\mathcal{D}}$, accomplished e.g. by some learning approach, is a crucial part of the ontology repair process. However, the formulation of a complete extension is outside the scope of this work where we focus on computing diagnoses. Following the approach suggested in [12], we approximate $EX_{\mathcal{D}}$ by the set $\bigcup_{p \in P} p$.
Example: Consider the OWL ontology $\mathcal{O}$ encompassing the following terminology $\mathcal{T}$:

$ax_1 : PhD \sqsubseteq Researcher$ $\quad$ $ax_4 : Student \sqsubseteq \neg DeptMember$
$ax_2 : Researcher \sqsubseteq DeptEmployee$ $\quad$ $ax_5 : PhDStudent \sqsubseteq PhD$
$ax_3 : PhDStudent \sqsubseteq Student$ $\quad$ $ax_6 : DeptEmployee \sqsubseteq DeptMember$

[1] Throughout the paper we consider debugging of inconsistent and/or incoherent ontologies, i.e. whenever not stated explicitly we assume $R = \{\text{consistency, coherency}\}$.

and an assertional axiom $\mathcal{A} = \{PhDStudent(s)\}$. Then $\mathcal{O}$ is inconsistent since it describes a PhD student as both a department member and not.

Let us assume that the assertion $PhDStudent(s)$ is considered as correct and is thus added to the background theory, i.e. $\mathcal{B} = \mathcal{A}$, and both sets P and N are empty. Then, the set of minimal diagnoses $\mathbf{D} = \{\mathcal{D}_1 : [ax_1], \mathcal{D}_2 : [ax_2], \mathcal{D}_3 : [ax_3], \mathcal{D}_4 : [ax_4], \mathcal{D}_5 : [ax_5], \mathcal{D}_6 : [ax_6]\}$ for the given problem instance $\langle \mathcal{T}, \mathcal{A}, \emptyset, \emptyset \rangle$. $\mathbf{D}$ can be computed by a diagnosis algorithm such as the one presented in [3].

With six diagnoses for six ontology axioms, this example might already give an idea that in many cases the number of diagnoses $\mathbf{D}$ can get very large. Without any prior knowledge, each of the diagnoses in $\mathbf{D}$ is equally likely to be the target diagnosis $\mathcal{D}_t$, that is selected by a user in order to formulate the intended ontology $\mathcal{O}_t := (\mathcal{O} \setminus \mathcal{D}_t) \cup EX_{\mathcal{D}_t}$. Identification of $\mathcal{D}_t$ can be accomplished by means of queries [12]. Thereby, the fact is exploited that ontologies $\mathcal{O} \setminus \mathcal{D}_i$ and $\mathcal{O} \setminus \mathcal{D}_j$ resulting in application of different diagnoses $\mathcal{D}_i, \mathcal{D}_j \in \mathbf{D}$ $(\mathcal{D}_i \neq \mathcal{D}_j)$ entail different sets of logical axioms.

Definition 3 (Query). *A set of logical axioms X_j is called a* query *iff there exists a set of diagnoses $\emptyset \subset \mathbf{D}' \subset \mathbf{D}$ such that X_j is entailed by each ontology in $\{\mathcal{O}_i^* \mid \mathcal{D}_i \in \mathbf{D}'\}$ where $\mathcal{O}_i^* := (\mathcal{O} \setminus \mathcal{D}_i) \cup \mathcal{B} \cup \bigcup_{p \in P} p$. Asking a query X_j to a user means asking them $(\mathcal{O}_t \models X_j?)$. The set of all queries w.r.t. $\mathbf{D}$ is denoted by $\mathbf{X}_\mathbf{D}$.*[2]

Each query X_j partitions the set of diagnoses $\mathbf{D}$ into $\langle \mathbf{D}_j^P, \mathbf{D}_j^N, \mathbf{D}_j^\emptyset \rangle$ such that $\mathbf{D}_j^P = \{\mathcal{D}_i \mid \mathcal{O}_i^* \models X_j\}$, $\mathbf{D}_j^N = \{\mathcal{D}_i \mid \mathcal{O}_i^* \cup X_j \text{ is inconsistent}\}$ and $\mathbf{D}_j^\emptyset = \mathbf{D} \setminus (\mathbf{D}_j^P \cup \mathbf{D}_j^N)$. If the answering of queries by a user u is modeled as a function $a_u : \mathbf{X} \to \{yes, no\}$, then the following holds: If $a_u(X_j) = yes$, then X_j is added to the positive test cases, i.e. $P \leftarrow P \cup \{X_j\}$, and all diagnoses in $\mathbf{D}_j^N$ are *rejected*. Given that $a_u(X_j) = no$, then $N \leftarrow N \cup \{X_j\}$ and all diagnoses in $\mathbf{D}_j^P$ are *rejected*. For the example ontology $\mathcal{O}$, we could, e.g., ask the user the query $X_1 := (\mathcal{O}_t \models \{DeptEmployee(s), Student(s)\}?)$ with the associated partition $\langle \mathbf{D}_1^P, \mathbf{D}_1^N, \mathbf{D}_1^\emptyset \rangle = \langle \{\mathcal{D}_4, \mathcal{D}_6\}, \{\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3, \mathcal{D}_5\}, \emptyset \rangle$. A negative answer would then eliminate $\{\mathcal{D}_4, \mathcal{D}_6\}$.

Definition 4 (Diagnosis Discrimination). ***Given*** *the set of diagnoses $\mathbf{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_n\}$ w.r.t. $\langle \mathcal{O}, \mathcal{B}, P, N \rangle_R$ and a user u,* ***find*** *a sequence $(X_1, \ldots, X_q)$ of queries $X_i \in \mathbf{X}$ with minimal q, such that $\mathbf{D} = \{\mathcal{D}_t\}$ after assigning $X_{i(i=1\ldots,q)}$ each to either P iff $a_u(X_i) = yes$ or N iff $a_u(X_i) = no$.*[3]

A set of queries for a given set of diagnoses $\mathbf{D}$ can be generated as shown in Algorithm 1. In each iteration, for a set of diagnoses $\mathbf{D}^P \subset \mathbf{D}$, the generator gets a set of logical descriptions X that are entailed by each ontology $\mathcal{O}_i^*$ where $\mathcal{D}_i \in \mathbf{D}^P$ (function GETENTAILMENTS). When we speak of entailments, we always address the output computed by the classification and realization services of a reasoner [1, p.323 ff.]. These axioms X are then used to classify the remaining diagnoses in $\mathbf{D} \setminus \mathbf{D}^P$ in order to obtain the partition $\langle \mathbf{D}^P, \mathbf{D}^N, \mathbf{D}^\emptyset \rangle$ associated with X. Then, together with its partition, X is added to the set of queries $\mathbf{X}$. Note that in real-world applications, investigation of all possible subsets of the set $\mathbf{D}$ might be infeasible. Thus, it is common to approximate

[2] For the sake of simplicity, we will use $\mathbf{X}$ instead of $\mathbf{X}_\mathbf{D}$ throughout this work because the $\mathbf{D}$ associated with $\mathbf{X}$ will be clear from the context.

[3] Since the user u is assumed fixed throughout a debugging session and for brevity, we will use a_i equivalent to $a_u(X_i)$ in the rest of this work.

Algorithm 1: Query Generation

Input: diagnosis problem instance $\langle \mathcal{O}, \mathcal{B}, P, N \rangle$, set of diagnoses $\mathbf{D}$
Output: a set of queries and associated partitions $\mathbf{X}$

```
foreach D^P ⊂ D do
    X ← getEntailments(O, B, P, D^P);
    if X ≠ ∅ then
        foreach D_r ∈ D \ D^P do
            if O*_r ⊨ X then D^P ← D^P ∪ {D_r};
            else if O*_r ∪ X is inconsistent then D^N ← D^N ∪ {D_r};
            else D^∅ ← D^∅ ∪ {D_r};
        X ← X ∪ ⟨X, D^P, D^N, D^∅⟩
return X;
```

the set of all minimal diagnoses by a set of *leading diagnoses*. This set comprises a predefined number n of minimal diagnoses.

The query generation algorithm returns a set of queries $\mathbf{X}$ that generally contains a lot of elements. Therefore the authors in [12] suggested two query selection strategies.

Split-in-half strategy, selects the query X_j which minimizes the scoring function $sc_{split}(X_j) = \big||\mathbf{D}_j^P| - |\mathbf{D}_j^N|\big| + |\mathbf{D}_j^\emptyset|$, i.e. this strategy prefers queries which eliminate half of the diagnoses independently of the query outcome.

Entropy-based strategy, uses information about prior probabilities for a user to make a fault in an axiom [12]. The fault probabilities of axioms $p(ax_i)$ can in turn be used to determine fault probabilities of diagnoses $\mathcal{D}_i \in \mathbf{D}$. The strategy is then to select the query which minimizes the expected entropy of the set of leading diagnoses $\mathbf{D}$ after the query is answered. This means that the expected uncertainty is minimized and the expected information gain is maximized. According to [7], this is equivalent to choosing the query X_j which minimizes the scoring function $sc_{ent}(X_j) = \sum_{a_j \in \{yes,no\}} p(a_j) \log_2 p(a_j) + p(\mathbf{D}_j^\emptyset) + 1$. After each query X_j, the diagnosis probabilities are updated according to the Bayesian formula [12].

A diagnosis discrimination procedure can use either of the strategies to identify the target diagnosis $\mathcal{D}_t$. The result of the evaluation in [12] shows that entropy-based query selection reveals better performance than split-in-half in most of the cases. However, split-in-half proved to be the best strategy in situations when only vague priors are provided, i.e. the target diagnosis $\mathcal{D}_t$ has rather low prior fault probability. Therefore selection of prior fault probabilities is crucial for successful query selection and minimization of user interaction.

Example (continued): In our example, if the user specifies $p(ax_{i\,(i=1,\ldots,4)}) = 0.001$, $p(ax_5) = 0.1$ and $p(ax_6) = 0.15$. Given $\mathcal{D}_t := \mathcal{D}_2$, the no-risk strategy sc_{split} (*three queries*) is more suitable than sc_{ent} (*four queries*) because the fault probabilities disfavor $\mathcal{D}_2$. If $\mathcal{D}_t := \mathcal{D}_6$, then the entropy-based strategy requires only *two queries* while it takes split-in-half *three queries* due to favorable fault probabilities.

We learn from this example that the best choice of discrimination strategy depends on the quality of the meta information in terms of prior fault probabilities. In cases where adequate meta information is not available and hard to estimate, e.g. Ontology Matching, the inappropriate choice of strategy might cause tremendous extra effort for the user interacting with the debugging system.

3 Risk Optimization Strategy for Query Selection

The proposed Risk Optimization Algorithm (RIO) extends entropy-based query selection strategy with a dynamic learning procedure that learns by reinforcement how to select optimal queries. Moreover, it continually improves the prior fault probabilities based on new knowledge obtained through queries to a user. The behavior of our algorithm can be co-determined by the user. The algorithm takes into account the user's doubt about the priors in terms of the initial cautiousness c as well as the cautiousness interval $[\underline{c}, \overline{c}]$ where $c, \underline{c}, \overline{c} \in [c_{\min}, c_{\max}] := [0, \lfloor |\mathbf{D}|/2 \rfloor / |\mathbf{D}|]$, $\underline{c} \leq c \leq \overline{c}$ and $\mathbf{D}$ contains at most n leading diagnoses (see Section 2). The interval $[\underline{c}, \overline{c}]$ constitutes the set of all admissible cautiousness values the algorithm may take during the debugging session. High trust in the prior fault probabilities is reflected by specifying a low minimum required cautiousness $\underline{c}$ and/or a low maximum admissible cautiousness $\overline{c}$. If the user is unsure about the rationality of the priors this can be expressed by setting $\underline{c}$ and/or $\overline{c}$ to a higher value. Intuitively, $\underline{c} - c_{\min}$ and $c_{\max} - \overline{c}$ represent the minimal desired difference in performance to a high-risk (entropy) and no-risk (split-in-half) query selection, respectively.

The relationship between cautiousness c and queries is formalized by the following definitions:

Definition 5 (Cautiousness of a Query). *We define the* cautiousness $caut(X_i)$ *of a query X_i as follows:*

$$caut(X_i) := \frac{min\left\{|\mathbf{D}_i^P|, |\mathbf{D}_i^N|\right\}}{|\mathbf{D}|} \in \left[0, \frac{\left\lfloor \frac{|\mathbf{D}|}{2} \right\rfloor}{|\mathbf{D}|}\right]$$

A query X_i is called braver *than query X_j iff $caut(X_i) < caut(X_j)$. Otherwise X_i is called* more cautious *than X_j. A query with highest possible cautiousness is called* no-risk query.

Definition 6 (Elimination Rate). *Given a query X_i and the corresponding answer $a_i \in \{yes, no\}$, the* elimination rate $e(X_i, a_i)$ *is defined as follows:*

$$e(X_i, a_i) = \begin{cases} \frac{|\mathbf{D}_i^N|}{|\mathbf{D}|} & \text{if } a_i = yes \\ \frac{|\mathbf{D}_i^P|}{|\mathbf{D}|} & \text{if } a_i = no \end{cases}$$

The answer a_i to a query X_i is called favorable *iff it maximizes the elimination rate $e(X_i, a_i)$. Otherwise a_i is called* unfavorable.

So, the cautiousness $caut(X_i)$ of a query X_i is exactly the minimal elimination rate, i.e. $caut(X_i) = e(X_i, a_i)$ given that a_i is the unfavorable query result. Intuitively, the user-defined cautiousness c is the minimum proportion of diagnoses in $\mathbf{D}$ which should be eliminated by the successive query. For braver queries the interval between minimum and maximum elimination rate is larger than for more cautious queries. For no-risk queries it is minimal.

Definition 7 (High-Risk Query). *Given a query X_i and cautiousness c, then X_i is called a* high-risk query *iff $caut(X_i) < c$, i.e. the cautiousness of the query is lower than the algorithm's current cautiousness value c. Otherwise, X_i is called* non-high-risk query. *By $HR_c(\mathbf{X}) \subseteq \mathbf{X}$ we denote the set of all high-risk queries w.r.t. c. For given cautiousness c, the set of all queries $\mathbf{X}$ can be partitioned in high-risk queries and non-high-risk queries.*

Given a user's answer a_s to a query X_s, the cautiousness c is updated depending on the elimination rate $e(X_s, a_s)$ by $c \leftarrow c + c_{adj}$ where $c_{adj} := 2\,(\overline{c} - \underline{c})\,adj$ denotes the cautiousness adjustment factor. The factor $2\,(\overline{c} - \underline{c})$ is a scaling factor that simply regulates the extent of the cautiousness adjustment depending on the interval length $\overline{c} - \underline{c}$. The more crucial factor in the formula is adj which indicates the sign and magnitude of the cautiousness adjustment.

$$adj := \frac{\left\lfloor \frac{|\mathbf{D}|}{2} - \epsilon \right\rfloor}{|\mathbf{D}|} - e(X_s, a_s)$$

where $\epsilon \in (0, \frac{1}{2})$ is a constant which prevents the algorithm from getting stuck in a no-risk strategy for even $|\mathbf{D}|$. E.g., given $c = 0.5$ and $\epsilon = 0$, the elimination rate of a no-risk query $e(X_s, a_s) = \frac{1}{2}$ resulting always in $adj = 0$. The value of ϵ can be set to an arbitrary real number, e.g. $\epsilon := \frac{1}{4}$. If $c + c_{adj}$ is outside the user-defined cautiousness interval $[\underline{c}, \overline{c}]$, it is set to $\underline{c}$ if $c < \underline{c}$ and to $\overline{c}$ if $c > \overline{c}$. Positive c_{adj} is a penalty telling the algorithm to get more cautious, whereas negative c_{adj} is a bonus resulting in a braver behavior of the algorithm.

The RIO algorithm, described in Algorithm 2, starts with the computation of minimal diagnoses. GETDIAGNOSES function implements a combination of hitting-set (HS-Tree) [11] and QuickXPlain [6] algorithms as suggested in [12]. Using uniform cost search, the algorithm extends the set of leading diagnoses $\mathbf{D}$ with a maximum number of most probable minimal diagnoses such that $|\mathbf{D}| \leq n$.

Then the GETPROBABILITIES function calculates the fault probabilities $p(\mathcal{D}_i)$ for each diagnosis $\mathcal{D}_i$ of the set of leading diagnoses $\mathbf{D}$ according to [12]. In order to take into account all information gathered by querying an oracle so far the algorithm adjusts fault probabilities $p(\mathcal{D}_i)$ as follows: $p_{adj}(\mathcal{D}_i) = (1/2)^z\, p(\mathcal{D}_i)$, where z is the number of precedent queries X_k for which $\mathcal{D}_i \in \mathbf{D}_k^{\emptyset}$. Afterwards the probabilities $p_{adj}(\mathcal{D}_i)$ are normalized. Note that z can be computed from P and N which comprise all query answers. This way of updating probabilities is exactly in compliance with the Bayes Formula [12]. Based on the set of leading diagnoses $\mathbf{D}$, GENERATEQUERIES generates all queries according to Algorithm 1. GETMINSCOREQUERY determines the best query $X_{sc} \in \mathbf{X}$ according to sc_{ent}. That is, $X_{sc} = \arg\min_{X_k \in \mathbf{X}}(sc_{ent}(X_k))$. If X_{sc} is a non-high-risk query, i.e. $c \leq caut(X_{sc})$ (determined by GETQUERYCAUTIOUSNESS), X_{sc} is selected. In this case, X_{sc} is the query with maximum information gain among all queries $\mathbf{X}$ and additionally guarantees the required elimination rate specified by c.

Otherwise, GETALTERNATIVEQUERY selects the query $X_{alt} \in \mathbf{X}$ $(X_{alt} \neq X_{sc})$ which has minimal score sc_{ent} among all least cautious non-high-risk queries L_c. That is, $X_{alt} = \arg\min_{X_k \in L_c}(sc_{ent}(X_k))$ where $L_c = \{X_r \in \mathbf{X} \setminus HR_c(\mathbf{X}) \mid \forall X_t \in \mathbf{X} \setminus HR_c(\mathbf{X}) : caut(X_r) \leq caut(X_t)\}$. If there is no such query $X_{alt} \in \mathbf{X}$, then X_{sc} is selected. Given the positive answer of the oracle, the selected query $X_s \in \{X_{sc}, X_{alt}\}$ is added to the set of positive test cases P or, otherwise, to the set of negative test cases N. In the last step of the main loop the algorithm updates the cautiousness value c (function UPDATECAUTIOUSNESS) as described above.

Before the next query selection iteration starts, a stop condition test is performed. The algorithm evaluates whether the most probable diagnosis is at least $\sigma\%$ more likely than the second most probable diagnosis (ABOVETHRESHOLD) or none of the leading diagnoses has been eliminated by the previous query, i.e.GETELIMINATIONRATE returns zero for X_s. In case that one of the stop conditions is fulfilled, the presently most likely diagnosis is returned (MOSTPROBABLEDIAG).

4 Evaluation

The main points we want to show in this evaluation are: On the one hand, independently of the specified meta information, RIO exhibits superior average behavior compared to entropy-based method and split-in-half w.r.t. the amount of user interaction required. On the other hand, we want to demonstrate that RIO scales well and that the reaction time measured is well suited for an interactive debugging approach.

As data source for the evaluation we used problematic real-world ontologies produced by ontology matching systems.[4] This has the following reasons: (1) Matching results often cause inconsistency and/or incoherency of ontologies. (2) The (fault) structure of different ontologies obtained through matching generally varies due to different authors and matching systems involved in the genesis of these ontologies. (3) For the same reasons, it is hard to estimate the quality of fault probabilities, i.e. it is unclear which of the existing query selection strategies to chose for best performance. (4) Available reference mappings can be used as correct solutions of the debugging procedure.

Matching of two ontologies $\mathcal{O}_i$ and $\mathcal{O}_j$ is understood as detection of correspondences between matchable elements of these ontologies. An ontology matching operation determines an *alignment* M_{ij}, which is a set of correspondences, i.e. tuples of the form $\langle x_i, x_j, r, v\rangle$, where $x_i \in Q(\mathcal{O}_i)$, $x_j \in Q(\mathcal{O}_j)$, $Q(\mathcal{O})$ is the set of matchable elements of an ontology $\mathcal{O}$, r is a semantic relation and $v \in [0, 1]$ is a confidence value. We call $\mathcal{O}_{iMj} := \mathcal{O}_i \cup M_{ij} \cup \mathcal{O}_j$ the *aligned ontology* for $\mathcal{O}_i$ and $\mathcal{O}_j$. In our approach the elements of $Q(\mathcal{O})$ are restricted to atomic concepts and roles and $r \in \{\sqsubseteq, \sqsupseteq, \equiv\}$ under the natural alignment semantics [8]

Example (continued): Imagine that our example ontology $\mathcal{O}$ evolved from matching two standalone ontologies $\mathcal{O}_1 := \{ax_1, ax_2\}$ and $\mathcal{O}_2 := \{ax_3, ax_4\}$ resulting in the alignment $M_{12} = \{ax_5, ax_6\}$. If we recall the set of diagnoses for $\mathcal{O}$ consisting of all single axioms in $\mathcal{O}$, we realize that the fault we are trying to find may be located either in $\mathcal{O}_1$ or in $\mathcal{O}_2$ or in M_{12}. Existing approaches to alignment debugging usually consider only the produced alignment as problem source. Our approach, on the contrary, is designed to cope with the most general setting: Any subset $S \subseteq \mathcal{O}_{1M2}$ of axioms of the aligned ontology can be analyzed for faults whereas $\mathcal{O}_{1M2} \setminus S$ can be added to the background axioms $\mathcal{B}$, if known to be correct. In this way, the search space for

[4] Thanks to Christian Meilicke for the supply of the test cases used in the evaluation.

Algorithm 2: Risk Optimization Algorithm (RIO)

Input: diagnosis problem instance $\langle \mathcal{O}, \mathcal{B}, P, N\rangle$, fault probabilities of diagnoses DP, cautiousness $C = (c, \underline{c}, \overline{c})$, number of leading diagnoses n to be considered, acceptance threshold σ
Output: a diagnosis $\mathcal{D}$

```
1  P ← ∅; N ← ∅; D ← ∅;
2  repeat
3      D ← getDiagnoses(D, n, O, B, P, N);
4      DP ← getProbabilities(DP, D, P, N);
5      X ← generateQueries(O, B, P, D);
6      X_s ← getMinScoreQuery(DP, X);
7      if getQueryCautiousness(X_s, D) < c then X_s ← getAlternativeQuery(c, X, DP, D);
8      if getAnswer(X_s) = yes then P ← P ∪ {X_s};
9      else N ← N ∪ {X_s};
10     c ← updateCautiousness(D, P, N, X_s, c, c_, c̄);
11 until (aboveThreshold(DP, σ) ∨ eliminationRate(X_s) = 0);
12 return mostProbableDiag(D, DP);
```

diagnoses can be restricted elegantly depending on the prior knowledge about $\mathcal{D}_t$, which can greatly reduce the complexity of the underlying diagnosis problem.

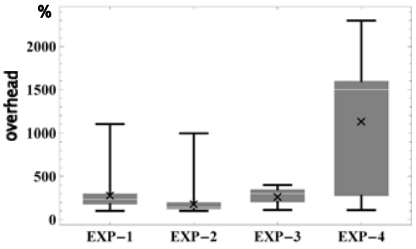
In [13] it was shown that existing debugging approaches suffer from serious problems w.r.t. both scalability and correctness of results when tested on a dataset of incoherent aligned OWL ontologies. Since RIO is an *interactive* ontology debugging approach able to query and incorporate additional information into its computations, it can cope with cases unsolved in [13]. In order to provide evidence for this and to show the feasibility of RIO – simultaneously to the main goals of this evaluation – we decided to use a superset of the dataset[5] used in [13] for our tests. Each incoherent aligned ontology $\mathcal{O}_{iMj}$ in the dataset is the result of applying one of the ontology matching systems COMA++, Falcon-AO, HMatch or OWL-CTXmatch to a set of six ontologies $Ont = \{\text{CRS}, \text{PCS}, \text{CMT}, \text{CONFTOOL}, \text{SIGKDD}, \text{EKAW}\}$ in the domain of conference organization. For a given pair of ontologies $\mathcal{O}_i \neq \mathcal{O}_j \in Ont$, each system produced an alignment M_{ij}. On the basis of a manually produced reference alignment $\mathcal{R}_{ij} \subseteq M_{ij}$ for ontologies $\mathcal{O}_i, \mathcal{O}_j$ (cf. [9]), we were able to fix a target diagnosis $\mathcal{D}_t$ for each incoherent $\mathcal{O}_{iMj}$. In cases where $\mathcal{R}_{ij}$ suggested a non-minimal diagnosis, we defined $\mathcal{D}_t$ as the minimum cardinality diagnosis which was a subset of $M_{ij} \setminus \mathcal{R}_{ij}$. In one single case, $\mathcal{R}_{ij}$ proved to be incoherent because an obviously valid correspondence $Reviewer_1 \equiv reviewer_2$ turned out to be incorrect. We re-evaluated this ontology and specified a coherent $\mathcal{R}_{ij}$. Yet this makes evident that, in general, people are not capable of analyzing alignments without adequate tool support.

In our experiments we set the prior fault probabilities as follows: $p(ax_k) := 0.001$ for $ax_k \in \mathcal{O}_i \cup \mathcal{O}_j$ and $p(ax_m) := 1 - v_m$ for $ax_m \in M_{ij}$, where v_m is the confidence of the correspondence underlying ax_m. Note that this choice results in a significant bias towards diagnoses which include axioms from M_{ij}. Based on these settings, in the first experiment (EXP-1), we simulated an interactive debugging session employing split-in-half (SPL), entropy (ENT) and RIO algorithms, respectively, for each ontology $\mathcal{O}_{iMj}$. Throughout all experiments, we performed module extraction before each test run, which is a standard preprocessing method for ontology debugging approaches. All tests were executed on a Core-i7 (3930K) 3.2Ghz, 32GB RAM and with Ubuntu Server 11.04 and Java 6 installed. The user-chosen parameters were set as follows: $|\mathbf{D}| := 9$ (proved to be a good trade-off between computational complexity for query generation and approximation of all minimal diagnoses), $\sigma := 85\%$, $c := 0.25$ and $[\underline{c}, \overline{c}] := [c_{\min}, c_{\max}] = [0, \frac{4}{9}]$. For the tests we considered the most general setting, i.e. $\mathcal{D}_t \subset \mathcal{O}_{iMj}$. So, we did not restrict the search for $\mathcal{D}_t$ to M_{ij} only, simulating the case where the user has no idea whether any of the input ontologies $\mathcal{O}_i, \mathcal{O}_j$ or the alignment M_{ij} or a combination thereof is faulty. In each test run we measured the number of required queries until $\mathcal{D}_t$ was identified. In order to simulate the case where the fault includes at least one axiom $ax \in \mathcal{O}_{iMj} \setminus M_{ij}$, we implemented a second test session with altered $\mathcal{D}_t$. In this experiment (EXP-2), we precalculated a maximum of 30 most probable minimal diagnoses, and from these we selected the diagnosis with the highest number of axioms $ax_k \in \mathcal{O}_{iMj} \setminus M_{ij}$ as $\mathcal{D}_t$ in order to simulate more unsuitable meta information. All the other settings were left unchanged. The queries generated in the tests were answered by an automatic oracle by means of the target ontology $\mathcal{O}_{iMj} \setminus \mathcal{D}_t$. The average metrics for the set of aligned ontologies $\mathcal{O}_{iMj}$ per matching system were as follows: $312 \leq |\mathcal{O}_{iMj}| \leq 377$ and $19.1 \leq |M_{ij}| \leq 28.4$.

[5] http://code.google.com/p/rmbd/downloads

	EXP-1	EXP-2	EXP-3	EXP-4
$q_{\text{SPL}} < q_{\text{ENT}}$	11%	37%	0%	29%
$q_{\text{ENT}} < q_{\text{SPL}}$	82%	56%	100%	71%
$q_{\text{SPL}} = q_{\text{ENT}}$	7%	7%	0%	0%
$q_{\text{RIO}} < \min$	4%	26%	29%	71%
$q_{\text{RIO}} \leq \min$	74%	74%	100%	100%

(a)



(b)

Fig. 1. (a) Percentage rates indicating which strategy performed best/better w.r.t. the required user interaction, i.e. number of queries. EXP-1 and EXP-2 involved 27, EXP-3 and EXP-4 seven debugging sessions each. q_{str} denotes the number of queries needed by strategy str and $\min$ is an abbreviation for $\min(q_{\text{SPL}}, q_{\text{ENT}})$. **(b)** Box-Whisker Plots presenting the distribution of overhead $(q_w - q_b)/q_b * 100$ (in %) per debugging session of the worse strategy $q_w := \max(q_{\text{SPL}}, q_{\text{ENT}})$ compared to the better strategy $q_b := \min(q_{\text{SPL}}, q_{\text{ENT}})$. Mean values are depicted by a cross.

In order to analyze the scalability of RIO, we used the set of ontologies from the ANATOMY track in the Ontology Alignment Evaluation Initiative[6] (OAEI) 2011.5, which comprises two input ontologies $\mathcal{O}_1$ (Human, 11545 axioms) and $\mathcal{O}_2$ (Mouse, 4838 axioms). The size of the alignments generated by 12 different matching systems was between 1147 and 1461 correspondences. Note that the aligned ontologies output by five matching systems, i.e. CODI, CSA, MaasMtch, MapEVO and Aroma, could not be analyzed in the experiments. This was due to a consistent output produced by CODI and the problem that the reasoner was not able to find a model within acceptable time (2 hours) in the case of CSA, MaasMtch, MapEVO and Aroma. Similar reasoning problems were also reported in [2]. Given the ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$, the output M_{12} of a matching system, and the correct reference alignment $\mathcal{R}_{12}$, we first fixed $\mathcal{D}_t$ as follows: Both ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ as well as the correctly extracted alignments $M_{12} \cap \mathcal{R}_{12}$ were placed in the background knowledge $\mathcal{B}$. The incorrect correspondences $M_{12} \setminus \mathcal{R}_{12}$ were analyzed by the debugger. In this way, we identified a set of diagnoses, where each diagnosis is a subset of $M_{12} \setminus \mathcal{R}_{12}$. From this set of diagnoses, we randomly selected one diagnosis as $\mathcal{D}_t$. Then we started the actual experiments: In EXP-3,[7] in order to simulate reasonable prior fault probabilities, a debugging session with parameter settings as in EXP-1 was executed. In EXP-4, we altered the settings in that we specified $p(ax_k) := 0.01$ for $ax_k \in \mathcal{O}_i \cup \mathcal{O}_j$ and $p(ax_m) := 0.001$ for $ax_m \in M_{ij}$, which caused the target diagnosis, that consisted solely of axioms in M_{ij}, to get assigned a relatively low prior fault probability.

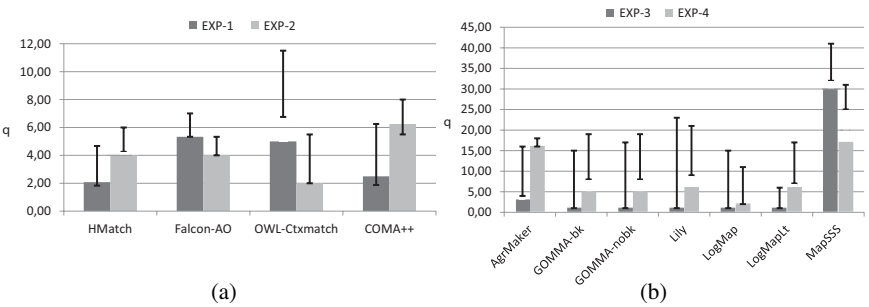
Results of both experimental sessions, ⟨EXP-1,EXP-2⟩ and ⟨EXP-3,EXP-4⟩, are summarized in Figure 2(a) and Figure 2(b), respectively. For the ontologies produced by each of the matching systems and for the different experimental scenarios, the figures show the (average) number of queries asked by RIO and the (average) differences to the number of queries needed by the per-session better and worse strategy of SPL and ENT, respectively. The results illustrate clearly that the average performance achieved by RIO was always substantially closer to the better than to the worse strategy. In both EXP-1 and EXP-2, throughout 74% of 27 debugging sessions, RIO worked as efficiently as the best strategy (Figure 1(a)). In more than 25% of the cases in EXP-2, RIO even

[6] http://oaei.ontologymatching.org

[7] For all details w.r.t. ⟨EXP-3,EXP-4⟩, see http://code.google.com/p/rmbd/wiki/ OntologyAlignmentAnatomy.

outperformed both other strategies; in these cases, RIO could save more than 20% of user interaction on average compared to the best other strategy. In one scenario involving OWL-CTXmatch in EXP-1, it took ENT 31 and SPL 13 queries to finish, whereas RIO required only 6 queries, which amounts to an improvement of more than 80% and 53%, respectively. In ⟨EXP-3,EXP-4⟩, the savings achieved by RIO were even more substantial. RIO manifested superior behavior to both other strategies in 29% and 71% of cases, respectively. Not less remarkable, in 100% of the tests in EXP-3 and EXP-4, RIO was at least as efficient as the best other strategy. Table 1, which provides the average number of queries per strategy, demonstrates that, overall, RIO is the best choice in all experiments. Consequently, RIO is suitable for both good meta information as in EXP-1 and EXP-3, where $\mathcal{D}_t$ has high probability, and poor meta information as in EXP-2 and EXP-4, where $\mathcal{D}_t$ is a-priori less likely. Additionally, Table 1 illustrates the (average) *overall* debugging time assuming that queries are answered instantaneously and the reaction time, i.e. the average time between two successive queries. Also w.r.t. these aspects, RIO manifested good performance. Since the times consumed by either of the strategies in ⟨EXP-1,EXP-2⟩ are almost negligible, consider the more meaningful results obtained in ⟨EXP-3,EXP-4⟩. While the best reaction time in both experiments was achieved by SPL, we can clearly see that SPL was significantly inferior to both ENT and RIO concerning the user interaction required and the overall time. RIO revealed the best debugging time in EXP-4, and needed only 2.2% more time than the best strategy (ENT) in EXP-3. However, if we assume the user being capable of reading and answering a query in, e.g., half a minute on average, which is already quite fast, then the overall time savings of RIO compared to ENT in EXP-3 would already account for 5%. Doing the same thought experiment for EXP-4, using RIO instead of ENT and SPL would save 25% and 50% of debugging time on average, respectively. All in all, the measured times confirm that RIO is well suited as *interactive* debugging method.

For SPL and ENT strategies, the difference w.r.t. the number of queries per test run between the better and the worse strategy was absolutely significant, with a maximum of 2300% in EXP-4 and averages of 190% to 1145% throughout all four experiments, measured on the basis of the better strategy (Figure 1(b)). Moreover, results show that the different quality of the prior fault probabilities in {EXP-1,EXP-3} compared to {EXP-2,EXP-4} clearly affected the performance of the ENT and SPL strategies (see first two rows in Figure 1(a)). This perfectly motivates the application of RIO.



(a) (b)

Fig. 2. The bars show the avg. number of queries (q) needed by RIO, grouped by matching tools. The distance from the bar to the lower (upper) end of the whisker indicates the avg. difference of RIO to the queries needed by the per-session better (worse) strategy of SPL and ENT, respectively.

	EXP-1			EXP-2			EXP-3			EXP-4		
	debug	react	q	debug	react	q	debug	react	q	debug	react	q
ENT	1860	262	3.67	1423	204	5.26	**60928**	12367	5.86	74463	5629	11.86
SPL	**1427**	**159**	5.70	**1237**	**148**	5.44	104910	**4786**	19.43	98647	**4781**	18.29
RIO	1592	286	**3.00**	1749	245	**4.37**	62289	12825	**5.43**	**66895**	8327	**8.14**

Table 1. Average time (ms) for the entire debugging session (debug), average time (ms) between two successive queries (react), and average number of queries (q) required by each strategy.

5 Conclusion

We have shown problems of state-of-the-art interactive ontology debugging strategies w.r.t. the usage of unreliable meta information. To tackle this issue, we proposed a learning strategy which combines the benefits of existing approaches, i.e. high potential and low risk. Depending on the performance of the diagnosis discrimination actions, the trust in the a-priori information is adapted. Tested under various conditions, our algorithm revealed an average performance superior to two common approaches in the field w.r.t. required user interaction. In our evaluation we showed the utility of our approach in the important area of ontology matching, its scalability and adequate reaction time allowing for continuous interactivity.

References

1. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.): The Description Logic Handbook: Theory, Implementation, Applications. Cambridge Press (2003)
2. Ferrara, A., Hage, W.R.V., Hollink, L., Nikolov, A., Shvaiko, P.: Final results of the Ontology Alignment Evaluation Initiative 2011. In: Evaluation (2011)
3. Friedrich, G., Shchekotykhin, K.: A General Diagnosis Method for Ontologies. In: Gil, Y., Motta, E., Benjamins, R., Musen, M. (eds.) The Semantic Web - ISWC 2005, 4th International Semantic Web Conference. pp. 232–246. Springer (2005)
4. Horridge, M., Parsia, B., Sattler, U.: Laconic and Precise Justifications in OWL. Proc of the 7th International Semantic Web Conference ISWC 2008 5318, 323–338 (2008)
5. Jiménez-Ruiz, E., Grau, B.C.: Logmap: Logic-based and scalable ontology matching. In: The Semantic Web - ISWC 2011. pp. 273–288. Springer (2011)
6. Junker, U.: QUICKXPLAIN: Preferred Explanations and Relaxations for Over-Constrained Problems. In: Proceedings of the 19th National Conference on AI, 16th Conference on Innovative Applications of AI. vol. 3, pp. 167–172. AAAI Press / The MIT Press (2004)
7. de Kleer, J., Williams, B.C.: Diagnosing multiple faults. Artif. Intell. 32(1), 97–130 (1987)
8. Meilicke, C., Stuckenschmidt, H.: An Efficient Method for Computing Alignment Diagnoses. In: Proceedings of the 3rd International Conference on Web Reasoning and Rule Systems. pp. 182–196. Springer-Verlag (2009)
9. Meilicke, C., Stuckenschmidt, H., Tamilin, A.: Reasoning Support for Mapping Revision. Journal of Logic and Computation 19(5), 807–829 (2008)
10. Noessner, J., Niepert, M., Meilicke, C., Stuckenschmidt, H.: Leveraging Terminological Structure for Object Reconciliation. The Semantic Web: Research and Applications pp. 334–348 (2010)
11. Reiter, R.: A Theory of Diagnosis from First Principles. Artif. Intell. 32(1), 57–95 (1987)
12. Shchekotykhin, K., Friedrich, G., Fleiss, P., Rodler, P.: Interactive ontology debugging : two query strategies for efficient fault localization. Web Semantics: Science, Services and Agents on the World Wide Web 12-13, 88–103 (2012)
13. Stuckenschmidt, H.: Debugging OWL Ontologies - A Reality Check. In: Proceedings of the 6th International Workshop on Evaluation of Ontology-based Tools and the Semantic Web Service Challenge (EON). pp. 1–12. Tenerife, Spain (2008)

Using the OM^{2}R Meta-Data Model for Ontology Mapping Reuse for the Ontology Alignment Challenge – a Case Study

Hendrik Thomas, Rob Brennan, Declan O'Sullivan

Federated Autonomic Management of End-to-end Communication Services (FAME),
Knowledge & Data Engineering Group, School of Computer Science and Statistics,
Trinity College Dublin, Ireland
{thomash, rob.brennan, declan.osullivan}@cs.tcd.ie

Abstract. Ontology matching and mapping is of critical importance to effective consumption of distributed and heterogeneous data-sets in today's Web of Data. Since 2004 the Ontology Alignment Evaluation Initiative (OAEI) provides a number of complex challenges to evaluate the performance of the increasing number of matching tools and methods. This leads to the question how the individual OAEI challenges and the individual alignment results can be documented best for effective online consumption, management and further analysis. In this paper, we argue that the current documentation of alignment creation lifecycle aspects within OAEI would benefit from more formal model support. In this paper we present a case study to show how our ontology-based meta-data model for ontology mapping reuse (OM^{2}R) can be applied for the OAEI to document alignment challenges and some quantification on the likely benefits in terms of helping challenge administrators and participants create consistent documentation in terms of high correctness and less inconsistent statements as well as results that are explicit, predictable and easy to interpret.

Keywords: Ontology Matching, Ontology Alignment, Meta-Data Model

1 Introduction

Ontology matching and mapping is of critical importance to effective consumption of distributed and heterogeneous data-sets in today's Web of Data [1,2]. To support the need for integration the number of methods that are being proposed for matching of ontologies/datasets has increased considerably, which consequently has created the need to establish a consensus for evaluation of these methods [2]. The Ontology Alignment Evaluation Initiative (OAEI) [3] organizes annual evaluation campaigns with the aim of "assessing strengths and weaknesses of alignment/matching systems; comparing performance of techniques; increase communication among algorithm developers" [4]. Each alignment challenge provides a collection of ontologies and reference alignments which enables a comprehensive evaluation of matching tools and their outputs in a controlled environment. In 2012 the OAEI provided seven distinct challenges and each challenge contains up to 58 individual alignment tasks. These challenges and reference ontologies are subject to changes from year to year to provide an even more effective and revealing test bed [3,5]. In the light of the OAEI's goals this leads to the question of how the individual OAEI challenges and the individual alignment results of the participants can be best documented for effective online consumption, management and analysis over time. In other words, for third parties to interpret and evaluate the alignment results of a particular matching method

correctly they often need to know precisely how each challenge was conducted. Also any changes to the challenge setups or target ontologies need to be documented clearly as the evaluation needs to be run over several years in order to allow for adequate measurements of the evolution of the field [3].

This creates the need for suitable documentation which can support participating users and researchers in evaluation of the alignment results [2,6]. The standards for such documentation tend to emerge over time as needs are identified and addressed. Since 2004 the OAEI has specified that each challenge must be documented on a specific web page to provide the scaffolding for the participants [4], e.g. including a short textual description of the dataset and evaluation modalities.[1] The majority of this information is provided in text form, lists and some embedded meta-data in the ontologies themselves. We argue that a more formal and structured model for the alignment lifecycle and appropriate alignment management meta-data may have benefits for both organisers and participants including the creation of more consistent documentation and the potential for automated re-use of alignments for other purposes in the future [2,7]. As each challenge is maintained by an independent group such a model can also be of benefit for the OAEI organisers to manage changes to reference alignments and to track submissions over the years to identify performance improvements and trends, e.g. to determine what alignment approaches are becoming more popular and more successful [3]. We argue that an improved meta-data model can help to leverage the experience gained in the OAEI to extend its focus from a pure test platform [8] to a large scale alignments repository [4] which can demonstrate how alignments can be managed, shared and reused over time successfully. To achieve such a shared understanding of matching challenges and the alignment creation in the true sense of the Semantic Web [9] a meta-data model needs to be documented clearly to help users understand the intended meaning of the individual fields easily [10,11]. To support analysis and reuse it needs to be formally detailed in a machine-interpretable notation such as OWL. It must promote the creation of consistent documentation instances in terms of correctness and avoidance of inconsistent statements.

In parallel to the work of OAEI, the authors have developed an ontology-based meta-data model for ontology mapping reuse (OM^2R) [7,12,13]. Thus OM^2R has a broader scope of supporting ontology mapping (alignment) management. Nonetheless at least part of the OAEI activity can be viewed as a very large-scale alignment management exercise, especially with respect to the historical result-sets. The challenge addressed in this paper is thus: can OM^2R be usefully applied to supporting OAEI activities and some quantification on the likely benefits in terms of helping challenge administrators and participants create consistent documentation in terms of high correctness and less inconsistent statements, experimental results that are explicit, predictable and easy to interpret. The model can also support matching retrieval and reasoning about matchings.

In this paper we present a case study to evaluate how the OM^2R model can be applied for the OAEI competition to document the alignment challenges to support machine-based online consumption, processing and further analysis of the submitted results through the publication of annotated OAEI challenges, data-sets and result-sets as linked data using the OM^2R vocabulary. In this first case study we have selected the benchmark dataset as a representative challenge from the OAEI initiative 2012 [4] and we will evaluate the individual meta-data fields proposed in OM^2R in relation to the current documentation.

[1] Please find more details on http://oaei.ontologymatching.org/doc/oaei-submitting.1.html

Please note the OM^{2}R was designed with a focus on ontology mappings but OAEI focuses on ontology alignment or matching [7,13]. In our terminology matchings are machine-generated correspondence candidates, an essential step in the creation of mappings which are confirmed correspondences created in the mapping phase as part of the overall ontology mapping creation lifecycle [14].

This paper is structured as follows. Section two gives a brief overview of other related meta-data models for ontology matchings. In section three we will provide a brief introduction to the OM^{2}R model. In section four we will discuss how OM^{2}R can be applied for the benefit of the OAEI initiative. The paper concludes with a summary and an outlook.

2 Related research

The need for a suitable meta-data model to document ontology matchings has been recognized in the current literature. For example J. Euzenat stated that one of the ten major challenges for ontology alignment is that management "must be complemented with rich metadata allowing users and systems to select the adequate alignments based on various criteria." [2,6] J. Euzenat and his team addressed this need by creating the ontology alignment format which offers a matching representation and basic meta-data identifying the addressed ontologies. Also an extended vocabulary [15] allows some meta-data to be embedded within the format.[2] In addition, EDOAL an expressive and declarative ontology alignment language extends the alignment format [22]. It provides a more detailed documentation of the matching algorithm elements but similar to the ontology alignment format it does not focus on the actual mapping creation lifecycle and management aspects.

Furthermore, we acknowledge the work of other authors in this area [16, 17]. For example N. Noy et al. proposed a community-driven ontology matching tool for public alignment reuse. This system annotates mapping elements in a given format but does not address the creation lifecycle or mapping reuse.

In addition, our work needs to be placed in context with ontology meta-data initiatives like the OMV (a meta-data model for ontologies and related entities [18]) or the PROV-DM (W3C data model for provenance interchange) [19]. These vocabularies can be used to express specific aspects of mappings efficiently like provenance, availability and statistics. Also important is the growing application of matchings in the linked data community to improve the interoperability between these still only loosely coupled data sets [16, 20]. The effort to distribute the matching creation tasks between different parties is increasing which implies the need for users to be able to assess the quality of matching and assess a possible reuse [4].

The current challenge for alignments management and therefore for the OAEI can be summarized as a need for a "convenient and interoperable support, on which tools […], can rely in order to store and share alignments. This involves using standard ways to communicate alignments and retrieve them. Hence, alignment metadata and annotations should be properly taken into account."[2].

The above discussed meta-data models demonstrate how other researchers have addressed these issues but their approaches are limited in the light of the OAEI documentation requirements as they are either focused on the representation of alignment correspondences and not on creation and management related meta-data data or the models are not specific and detailed enough for the alignment management

[2] More information can be found on: http://alignapi.gforge.inria.fr/labels.html

and reuse. The OM^{2}R can benefit from their contributions but we argue that the wider objective OM^{2}R which focuses on the whole ontology matching and mapping creation lifecycle can better support the creation of documentation to support retrieval, management and analysis over time for the OAEI.

3 Overview of OM^{2}R

3.1 Basic principles

The main design objective of OM^{2}R was to create a meta-data model for ontology mappings which covers the complete lifecycle including the matching phase to support mapping discovery and management [7,12]. Various formats are available to document ontology matching and mappings [12]. The design of a mapping representation which fulfils all possible requirements for expressing the correspondences might be overly complex, hard to enforce consistency on or alternatively represent only the lowest common denominator information [2, 12]. In contrast, a meta-data layer which documents the mapping lifecycle can complement existing mapping representations. Thus OM^{2}R is used to provide a common vocabulary for documenting mappings but is kept distinct from the mappings themselves. Hence OM^{2}R does not replace existing mapping representation languages but it compliments them with extensive lifecycle and context information which references the actual alignment themselves in a language neutral way. OM^{2}R meta-data is intended to be shared between users and applied in different contexts. Thus unambiguous meaning in terms of a shared common understanding of the documentation fields is essential. Hence OM^{2}R is expressed in an ontology which describes the meta-data structures and embeds extensive descriptions of the model elements (e.g. a short name, a definition, acronyms and a unique identifier) inside the actual model. The ontology contains 38 classes and 21 typed object relations between the individual meta-data fields which can be interpreted by editors, e.g. to enable highlighting of compatible field options. OWL-DL was used to model OM^{2}R instead of RDF(S) because it provides the necessary expressivity and supports greater reasoning to reveal implicit knowledge [7]. In our view, the key to understanding how a particular mapping was created lies in the ontology mapping lifecycle. In other words, the individual phases of the life cycle are used as the basis for the structure of the OM^{2}R and the involved activities provided an indication of what aspects need to be documented in meta-data fields. A common agreement on the phases involved in a full ontology mapping lifecycle has not yet emerged [7,14]. Please find below a mapping lifecycle proposal based on [14] which was used for OM^{2}R:

1.) **Characterisation phase:** The focus of this phase is the discovery of the ontologies which are subject of the mappings in term of the identification of the ontologies and their nature with respect to their amenability for matching methods.
2.) **Matching phase**: The objective of this phase is the description of identification of mapping candidates, either identified by manual selection or by automated matching algorithms [9,20].
3.) **Mapping phase:** The third stage involves the generation of information necessary for the execution of mappings as well as the creation of confirmed mappings.
4.) **Execution phase:** The identified committed and approved mappings can then be rendered into different mapping formats in order to enable processing and sharing.
5.) **Management phase:** Ontology mappings generated in the previous phases need to be managed and maintained until their withdrawal. This includes the sharing of mapping information with third parties, the integration of mapping into other mapping applications.
6.) **Meta-Data creation:** Conceptually a parallel activity to the phases above where meta-data is collected and processed, e.g. automatically extracted from ontologies or manually entered by involved stakeholders. Appropriate tool support may integrate it into the other lifecycle phases.

The key contribution of the formal OM^{2}R model is that it can support the creation of consistent documentation that is suitable for automated consumption and processing. More specifically our model can contribute to the following consistency aspects [21]: structural consistency, logical consistency and application consistency. Each is described in more detail below.

Structural Consistency ensures that the ontology obeys the constraints of the ontology language with respect to how the constructs of the ontology language are used [21]. The OM^{2}R model provides a common set of concepts and relations, thus a clear documented template allowing two users to express their facts by using the same vocabulary and semantic.

Logical consistency sees the ontology as a logical theory, which considers an ontology as logically consistent if it does not contain contradicting information [21]. By explicitly modelling allowed and appropriate relationships, the OM^{2}R model contains information about compatible relations between meta-data fields. For example if an ontology was expressed in the notation RDF/XML and in the formal language RDF(S), this reflect a compatible relation between the notation and formal language used which is modelled explicitly in the OM^{2}R. Our mapping documentation tool based on OM^{2}R can use these relations to highlight logical consistent options in the UI to support the editing process.

Application consistency relates to aspects not captured by the underlying ontology language itself, but rather given by some application or usage context [21]. In our context this relates to the ability of OM^{2}R to support the actual correctness of documentation in relation to a given matching and mapping management scenario.

The actual OM^{2}R model is available for download.[3] Please note beside the OWL file we provide on the same page the Protegé project files which enables you to start using the model to document your own matchings straightforward.

2.2 Evaluation

To validate the OM^{2}R we conducted a wide-scale end-user evaluation experiment with 50 participants drawn from the semantic web research community in 2010. The hypothesis was that the proposed OM^{2}R fields and their structure are considered relevant by users for a mapping reuse decision. The participants were given two mapping documentation scenarios and could rate the relevance of the individual fields for documentation and a reuse decision. The data showed that information identifying the addressed ontologies and matchings (e.g. names and location) are considered most relevant closely followed by details about the specific matching and mapping process used. Overall all of the 29 meta-data fields were considered relevant[4].

In 2012 we conducted a more practical task-oriented experiment with the hypothesis that OM^{2}R can support the creation of consistent documentation (see section 2.2) of the ontology mapping lifecycle and is usable by novice and experienced users in ontology mappings. The users were presented with an editing interface based on the OM^{2}R and asked to document the identification and matching phase of a sample matching scenario based on textual instructions. We used precision and recall [10] as an indicator for the level of achieved application and logical consistency. Overall 48 users completed the experiment with a ration of 40% experts with previous matching experience and 60% novice users with no experience. The following table shows the data we collected:

[3] The OM^{2}R model can be downloaded from: http://www.modelmapping.org/om2r

[4] The % of users who rated a field as relevant ranged from 77% to 23% with a mean of 60%

Metric	All Participants	Expert users	Novice
Application – recall	78 %	78.5 %	77.6%
Application – precision	81.8 %	79.1%	83.6 %
Logical- recall	86 %	91 %	82.2 %
Logical – precision	85 %	85.8%	84.6 %

Tab. 1 Average metrics for application and logic consistency

This evidence supports our claim that the OM^{2}R can support users in the creation of consistent documentation. Also we could not find any statistically significant difference between the support for experts and novice users.

4 Application of OM^{2}R to the OAEI

In this section we discuss the current documentation provided by the OAEI and show how the OM^{2}R can help to add an additional beneficial documentation layer.

4.1 General Approach

To show the benefits of the OM^{2}R an understanding of the involved stakeholders is needed. Please find below an overview:

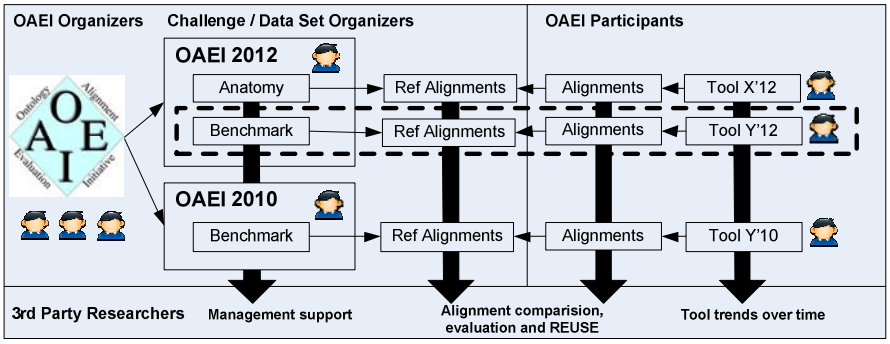


Fig. 1 Overview of the OAEI Stakeholders

The first involved group are the OAEI organizers which are responsible for the overall management, the submissions and the publication of the results for each OAEI initiative per year. Each individual challenge is maintained by an independent group who manages the different alignment tasks, ontologies and reference alignments. Also involved are the actual participants who use their matching tools to complete the individual tasks by submitting alignments or since 2011 their applications as a bundle. The fourth stakeholders are 3rd party researchers, who utilize the results published by the OAEI committee to learn more about the performance of the matching tools based on a metric approach [2]. We argue that an analysis of the reference ontologies, the actual alignments created by the participants as well as the provided reference alignments are of similar interest and value.

The current documentation provided by the OAEI is focused on individual challenges and the different initiatives per year. Each challenge is documented on a specific web page. This web page represents the main documentation source and provides the participants with the needed information to join the challenge. The primary focus is on online consumption as the majority of information is presented in text form, tables and some few meta-data fields embedded in the reference ontologies

and alignments. The dotted line in figure 1 indicates the addressed stakeholder of this horizontal documentation focus.

We argue that the OM^{2}R can provide an additional meta-data layer which can extend the current documentation with a more formal model to address the particular needs of 3rd party researchers and organizers. OM^{2}R allows users to create more consistent (see section 3.2), easier to interpret and more explicit documentation which can help to identify trends easier as well as an enable a more detailed comparison of the results of individual contributors over time. We argue that the OM^{2}R can bring the current available information together, add more structure combined with a higher level of detail and a time dimension (big black arrows in fig 1). This can help OAEI organizers and 3rd party researcher to keep a better overview and to manage changes of data sets over time.

To achieve this objective, the OM^{2}R uses a different representation approach for meta-data. Instead of a focus on text designed for human consumption it focused on retrieval and automated processing. It targets specifically the objects of interest for matching embedded in a lifecycle structure. The OM^{2}R is expressed as an ontology and therefore all meta-data information are stored as explicit and meaningful triples, e.g. om2:source_ontology hasNotaton rdf/xml = object of interest - typed relation - meta-data field option. Also explicit relations between the field options are included in the model, e.g. compatible relation between language and notation. This rich index structure makes the editing and the interpretation of the intended meaning easier, less ambiguous and provides a better structure for human and automated consumption. Also the current documentation is limited to single data sets per initiate. This is well suited for challenge participants but limits the view for researchers and organizers. The benefit of the OM^{2}R is that multiple alignments can be documented in one OM^{2}R model. This is particular relevant for the benchmark data set which is designed to be stable over time but as the web page points out the reference ontology has changed in 2010. Comparison, retrieval and reasoning can be supported better if the reference ontologies and their individual alignment versions per year could be documented in one OM^{2}R.

4.2 Meta-Data overview

To gain a more detailed understanding of the individual meta-data that is typically provided in OAEI we will focus in the following sections on one representative alignment challenge. More specifically we demonstrate the contribution of the OM^{2}R for the OAEI by focusing on the meta-data provided for the characterisation and matching phase of the lifecycle.

The selected challenge needs to be extensive in order to provide sufficient context for documentation and was used in previous OAEI initiative in order to allow a comparison of the available meta-data over time. In the latest OAEI challenge in 2012 the following data sets were provided: Benchmark, Anatomy, Conference, Multifarm, Library, Large Biomedical Ontologies, Instance matching [4]. If we consider the last four OAEI challenges (year 2012, 2011.5, 2011, 2010), only the following data sets have been used in all four challenges: Benchmark, Anatomy, Conference. If we compare the provided documentation for the 2012 challenge we can see, that the documentation webpage for the benchmark data set contains the most detailed documentation and therefore the highest amount of meta-data information.[5] It can therefore provide the most insight and will be the focus of our discussion.[6]

[5] The word count for the benchmark page was 3505, for anatomy 702 and for conference 544.
[6] Please see for details: http://oaei.ontologymatching.org/2012/benchmarks/index.html.

The following table provides an overview of the individual meta-data fields which have been rated by our end user experiments as relevant (see section 3.2) for the identification, characterisation and matching phase. It shows which information are provided by the OAEI and the corresponding fields in the OM^{2}R. Please note the column "OAEI Fields" indicates if the meta-data information is presented by the OAEI in an explicit field (e.g. embedded in the ontology) or was mentioned in an unstructured text segment. The column also tells you if the information is available for all (A) addressed target and source ontologies or only for some (S). Following the table the individual lifecycle phases are discussed in more detail:

Meta-Data field	OAEI Fields	OM^{2}R - Meta-Data Fields
Name of ontologies	Text (A) Field (S)	SourceOntology :Om2r:human_readable_name: "Biology Top Level Ontology"
Description of ontologies	Text (S) Field (S)	Om2r:description
Location of ontology	Text (A)	Om2r:hasLocation (type url)
Creation date of ontologies	Field (S)	Om2r:hasCreationDate (type date)
Unique identifier for ontologies	Field (A)	Om2r:hasIdentifier
Ontology Version	Missing	Om2r:hasVersion (URI)
Complexity of the ontology	Text (S)	Om2r:hasClassCount 73, hasInstanceCount 3 hasPropertyClass 3
Design of the ontologies	Text (S)	Om2r:hasDesign om2r:deep_hierarchy.
Notation of Ontologies	Text (S)	Om2r:hasNotation RDF/XML
Formal Language of Ontologies	Text (S)	Om2r:hasFormalLangauge OWL
Matching Location	Text (A)	Matching Om2r:hasLocation: www (URL)
Formal Language of the Matching	Test (S)	Om2r:hasformalMatchingLanguage: EDOAL
Notation of the Matching	Missing	Om2: hasNotation: RDF/XML.
Matching Method	Missing	Om2r:hasMethod (manual, automatic, mixed)
Matching Tool	Missing	Om2r:isTool AlignmentServer
Matching Algorithm	Missing	Algorithm :encodedIn: Java, Algorithm :hasJavaClass: org.stringComp, Algorithm :hasSource: freecode.org/a.zip
Algorithm is based on	Missing	Om2r:isBasedOn rdfs:label, rdfs:class
Applied Threshold	Missing	Om2r:has_Applied_Threshold
Matching Scope	Missing	Om2r:hasScope (complete or partial)
Matching Requirements	Missing	Om2r:hasMatchRequirements (text)

Tab. 2 Comparison of OM^{2}R meta-data fields with the OAEI

In the following sections we will discuss the provided meta-data in more detail. However space is limited here and it recommended that readers download the OM^{2}R ontology for themselves (see footnote 3) and use their preferred tool to explore it.

4.3 Phase 1.1 - Identification of the addressed ontologies

To begin a challenge a participant requires details about the addressed ontologies. On the web page of the benchmark data set a brief description is provided for the source ontology which is referred to as "reference ontology" but also as "bibliographic ontology" and in the task section as "test". Furthermore the web page lists 58 specific tasks where the target ontology is specified, for example [4]:

> 104) Concept test: Language restriction – This test compares the ontology with its restriction in OWL Lite (where unavailable constraints … Ontology : [RDF/XML] [HTML] Alignment : [RDF/XML]
> 201[-2-4-6-8]) Systematic: No names - Each label or identifier is replaced by a random one. Ontology : [RDF/XML] [HTML] Alignment : [RDF/XML] [HTML]

Please note the amount of descriptive information for the target ontologies is not consistent for each task, e.g. see example for test 104 vs. 201. Please note that the tasks listed on the lower part of the page contain less information than on the top. Some of the target ontologies have additional meta-data embedded in their source code, e.g. <dc:description>, <rdfs:label> for task 225 but these information can not be

found consistently, e.g. are missing for task 250 and 303. The provided alternative names and descriptions are quite suitable for participants. However, a more consistent approach is needed to support retrieval, analysis and automatic processing. Thus the OM^{2}R provide the following explicit fields: (**Target and Source) Ontology Name** and **(Target and Source) Ontology Description field.** Thanks to the ontology approach additional meta-data can be expressed easily and meaningful, e.g. hasAlternativeName e.g. hasNaturalLanguage "German".
In addition, the data set provides information where the sources of the addressed ontologies can be downloaded. The OM^{2}R provides similar information but in an explicit field **(Target and Source) Ontology Location** to allow automated system to retrieve the required information which currently can be difficult, e.g. the source for the reference ontology points to a section on the web page rather to the actually file.

To track down changes of reference ontologies over time or to negotiate a possible reuse of an ontology it is essential to be able to contact the authors. Currently only few contact information are embedded in some of the reference ontologies, e.g. <dc:contributor>Antoine Zimmermann antoine.zimmermann@inrialpes.fr </dc:contributor>). To promote the publication of such information, the OM^{2}R provides a dedicated field for this purpose: **(Target and Source) Ontology Editor**. Please note, to simplify the population existing ontology templates for contact details can be used in the OM^{2}R to help identify the creator more accurately, e.g. FOAF: Ontology creator om2r:firstName Hendrik, Ontology creator om2r:surname Thomas.

For an analysis over time information about the current version of the ontologies and their changes are critical. A good indicator is the creation time and the OAEI provides some textual references. As various date formats exist, an explicit and unambiguous representation is helpful to avoid confusion which why the OM^{2}R provides the field **(Target and Source) Ontology Creation Date** for an explicit time and date of the creation of the ontology. Internally the date will be represented as a set of explicit triples: CreationDate :hasYear: 2010, CreationDate :hasMonth: 5, CreationDate :hasDay: 4, CreationDate :hasTimeZone: MEZ.

Another relevant aspect is specific information about the changes to reference ontologies as they can create a bias for comparison of results over time. For example, in 2012 the web page states that the reference ontology for the benchmark data set has been altered and "The test is not anymore based on the very same dataset that has been used from 2004 to 2010. We are now able to generate undisclosed tests with the same structure. They provide strongly comparable results and allow for testing scalability." [4] but no further details are provided. The OM^{2}R can assist in providing a more detailed and structured documentation of changes with:

Ontology Version provides details about the specific version entered by the editor and a simple hashId to enable an automated and unbiased check for differences: om2r:asVersionId and om2r:hasHashID. Also the **Ontology Change Log** fields can contain elements with a short textual description of the specific conducted changes.

For humans names are a dominant key for identification but in the Semantic World an unambiguous identifier for the ontologies is essential to allow automated processing. In the data set the base url of each ontology is used for this purpose which is unique for each challenge and each data set, e.g. <rdf:RDF xml:base="http://oaei.ontologymatching.org/2012/ benchmarks/ 250/onto.rdf#"> for the task 250. Till 2010 the web page claims the same ontology was used for this dataset but each ontology has a unique identifier and is therefore potentially different. To avoid any miss interpretations the OM^{2}R provides an explicit field **Ontology Identifier** where unique identifier can be stored.

4.4 Phase 1.2 – Characterisation of the addressed ontologies

Information about the language aspects of the ontology files are of crucial importance for processing and compatibility issues of editing tools. The OAEI provides information about notation and the formal language in text form on the data set page, e.g. the web page states that the reference ontology is available in rdf/xml. The formal language is mentioned in the text but not consistently and in some cases missing, e.g. see the example description for task 236 in section 4.3. To help users in interpreting and reusing the provided resources more explicit information can be helpful, e.g. reasoning can only be applied to OWL DL not OWL Lite, thus stating the language as OWL would be too broad. OM^{2}R addresses this issues with the following fields: **Ontology Formal Language:** An ontology language is a formal language used to encode the ontology. As there are a number of such languages this field specifies the language, :hasFormalLanguage: http://www.w3.org/2002/07/owl. In case of OWL it is important to specify the sublanguage, too e.g. :subLanguage: OWL-DL. **Ontology Notation**: Beside the ontology language, the specific exchange notation used to represent the addressed ontology can be specified which is essential for tool support and exchange, e.g. TargetOntology :hasNotation: RDF/XML.

The next relevant area for 3rd party researchers and participants is the complexity of the addressed ontologies which is an essential factor when choosing an appropriate algorithm. For this purpose the OAEI provides information about the size of the source ontology (e.g. number of classes) but only in text form and not for the target ontologies. To support analysis and to judge the performance results in relation to the complexity more explicit fields can be helpful to allow automatic harvesting and processing. The OM^{2}R can assist the publication of these information with the following fields for the target and the source ontology:

Ontology Size: An explicit statement of the amount of classes, properties and instance, e.g. om2r:TargetOntology om2r:containsAmountOfClasses: 50

Ontology Design: Provides an indication of the basic design of the ontology, e.g. a sophisticated and deep hierarchy, a flat class hierarchy with few parent-client classes. The motivation for this field is to provide a broad classification, as different matching algorithms are more suitable for certain structures and size information alone are not sufficient enough, e.g. om2r:TargetOntology om2r:hasDesign om2r:flat_hierarchy.

4.5 Matching phase

The next area of relevance which was identified in our studies (see section 3.2) are details about the matching representation. This refers to the provided gold standard per dataset task and the individual submissions of the participants. The OAEI provides a location where the alignment can be downloaded.[7] In the OM^{2}R we provide the following explicit field for the location: **Matching Ontology Location.** This is a URL where the file can be downloaded, e.g. Matching :hasLocation.

In regards to the language aspect only the description text per task indicates that the alignment is expressed in XML/RDF but no information are provided for the formal language, e.g. EDOAL. In the OM^{2}R we provide the fields: **Matching Language:** A matching language is a formal language used to encode the correspondences, e.g. :hasFormalMatchingLanguage: om2r:edoal. In addition, the

[7] Please note we observed an inconsistency in regards to provenance and location, as in the 2012 challenge the alignment links in task 104 points to the 2011 challenge http://oaei.ontologymatching.org/2011/benchmarks/104/refalign.rdf

specific exchange **Matching Notation** can be specified which is essential for tool compatibility and reuse, e.g. Matching :hasNotation: RDF/XML.

Another key aspect are details about the actual method used to created the alignments. We can note that for all 58 tests a gold standard reference alignment is provided but most of the representations do not provide any information about the method or tool used to generate them. The alignment format provides a corresponding meta-data field like <method> for information on the applied matching class but none of this information have been provided in OAEI 2012 benchmark data set.

The OM^{2}R support the population of these information with the following fields. Please note the OM^{2}R provide specific instances for all fields which a user can select during the editing process and for each field option the compatible options in related fields are documented, e.g. compatible matching tools for matching methods.

Matching Method: Which generic method was used to find suitable candidates for a matching in the addressed ontologies? Om2r:hasMethod – manual, automatic, mixed
Matching Tool: Specified the tool which was used to generate the alignment, .e.g. hasMatchingTool
Matching Algorithm: If an automated selection was applied, this section provides a descriptive, human-readable label to identify the matching algorithm used. For example: matching :basedOn: Levenshtein distance, Levenshtein distance :isDefinedIn: http://en.wikipedia.org/wiki/Levenshtein_distance
Matching Algorithm Implementation: A descriptive, human-readable label to identify the specific implementation of the algorithm. Could be a URL or a specific JAVA class name like org.jena.stringComparsion. Also helpful is to provide a URL to download the source code. For example: Algorithm :encodedIn: Java, Algorithm :hasJavaClass: org.jena.stringComparsion, Algorithm :hasSource: http://www.freecode.org/123.zip
Applied Threshold: Defines the specific value of the similarity measure which needs to be passed in order to justify a matching pair based on the assumptions of the individual algorithm, e.g. om2r:has_Applied_Threshold. More complex methods may need multiple thresholds or iterations to be modeled instead.
Matching Scope: Defines the scope or area the matching is applied. In particular if all elements are matched to each other or only a particular subset, e.g. om2r:hasScope – complete or partial
Element Matching is based on: Defines the elements which are analyzed by the algorithm to identify the matching pairs, e.g. RDFSLabelForClass
Matching Requirements: Provides details of the specific requirements which needed to be fulfilled to apply the matching, e.g. hasMatchRequirements (text)

5 Conclusions and Final Remarks

In this paper we presented a case study to show how our ontology-based meta-data model for ontology mapping reuse (OM^{2}R) can be used to extend the current documentation of the OAEI for alignment challenges. We showed how the OM^{2}R can help administrators and participants create more consistent documentation instances in terms of high correctness and less inconsistent statements as well as support 3rd party researchers with more explicit, detailed, predictable and easy to interpret documentations. We argue that an improved meta-data model can help to leverage the experience gained in the OAEI to extend its focus from a pure test platform [8] to a large scale alignments management repository [4] which can demonstrate how alignments can be managed, shared and reused over time successfully. The overall objective of the OM^{2}R is to support the sharing of a common understanding of the ontology matching creation and application lifecycle which can hopefully provide a positive contribution to promote and support the reuse of alignments outside the current testing scope.

Acknowledgement

This work is partially funded through the Science Foundation Ireland FAME Strategic Research Cluster (award No. 08/SRC/I1408), http://www.fame.ie.

References

1. Marshall S. et al: Emerging practices for mapping and linking life sciences data using RDF - a case series. In: *Journal of Web Semantics: Science, Services and Agents*, Volume 12, 2012.
2. Shvaiko, P., Euzenat, J.: Ontology matching: state of the art and future challenges. In *Journal of IEEE Transactions on Knowledge and Data Engineering*, 2012 in press.
3. Euzenat, J., Meilicke, C., Shvaiko, P. et al.: Ontology Alignment Evaluation Initiative: six years of experience, In: *Journal on Data Semantics,* Volume XV Edition 6720, pp 158-192, 2011.
4. Euzenat, J.: Ontology Alignment Evaluation Initiative, main homepage, 2012 http://oaei.ontologymatching.org/
5. Euzenat J. et all: Final results of the Ontology Alignment Evaluation Initiative 2011, In: *The Sixth International Workshop on Ontology Matching: Proceedings of the 10th International Semantic Web Conference* ISWC-2011, 2011.
6. Shvaiko, P., Euzenat, J.: Ten Challenges for Ontology Matching. In*: Proceedings of the 7th International Conference on Ontologies, DataBases, and Applications of Semantics* (ODBASE), 2008.
7. Thomas, H., Brennan, R. O'Sullivan, D.: MooM - a Prototype Framework for Management of Ontology Mappings, In: *Proceedings of the 25th IEEE International Conference on Advanced Information Networking and Applications*, Singapore, 22-25 March, 2011, IEEE, pp 548 – 555.
8. Euzenat, J. et al: Results of the Ontology Alignment Evaluation Initiative 2011, In: *Proceeding of the 6th ISWC workshop on ontology matching*, pp 85-110, 2011.
9. Gruber, T. R.: Towards Principles of Ontologies Used in Knowledge Sharing. In: *International Journal of Human-Comper Studies*, Nr. 43, pp. 907-928, 1994.
10. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. New York, NY: ACM Press, Addison-Wesley, pp. 75 – 78, 1999.
11. Fugmann, R.: Subject Analysis and Indexing: Theoretical Foundation and Practical Advice. Frankfurt a. M., Indeks, 1993.
12. Thomas, H., O'Sullivan, D., Brennan, R.: Ontology Mapping Representations: a Pragmatic Evaluation. In: *21st International Conference on Software Engineering and Knowledge Engineering,*. SEKE 2009, 1 - 3 July 3, Boston, 2009, pp. 228 - 232.
13. Feeney, K., Brennan, R., Keeney, J., Thomas, H., Lewis, D., Boran, A., O'Sullivan, D., Enabling Decentralised Management through Federation, In: *Journal Elsevier Computer Networks*, Volume 54, Issue 16, November 2010.
14. O'Sullivan, D., Wade, V., Lewis, D. Understanding as We Roam, In: *IEEE Internet Computing Journal*, Volume 11, Issue 2, 2007, pp. 26 – 33.
15. Euzenat, J.: *Alignment API and server*, (version 3.2) https://gforge.inria.fr/docman/view.php/117/5036/alignapi.pdf, 2008.
16. N. Noy, N. Griffith, and M. Musen. Collecting community-based mappings in an ontology repository. In *Proc. of International Semantic Web Conference (ISWC)*, Karlsruhe, Germany, 2008.
17. Ghazvinian, A., Noy, N. F. , Jonquet, C. et al : What Four Million Mappings Can Tell You About Two Hundred Ontologies, In: *Proceedings of International Semantic Web Conference (ISWC)*, Washington DC 2009.
18. Palma, R. Hartmann J, Hasse P.: Documenation Report, http://surfnet.dl. sourceforge.net/project/omv2/OMV%20Documentation/OMV-Reportv2.4.1.pdf, 2009.
19. Moreau, L., Missier, P: PROV-DM: The PROV Data Model, W3C Working Draft, http://www.w3.org/TR/prov-dm/, July 2012.
20. Millard, I., Glaser, H., Salvadores, M. and Shadbolt, N.: Consuming multiple linked data sources: Challenges and Experiences. In: *First International Workshop on Consuming Linked Data (COLD2010)*, Shanghai, 2010.
21. Haase, P., Stojanovic, L.: Consistent Evolution of OWL Ontologies. In proceeding of the 5th European Semantic Web Conference, pp. 182-197, 2005.
22. David, J., Euzenat, J., Scharffe, F., Trojahn dos Santos, C.: The Alignment API 4.0. In: Semantic web journal Volume 2, Nr.1 pp. 3-10, 2011.

Results of the Ontology Alignment Evaluation Initiative 2012*

José Luis Aguirre[1], Kai Eckert[4], Jérôme Euzenat[1], Alfio Ferrara[2], Willem Robert van Hage[3], Laura Hollink[3], Christian Meilicke[4], Andriy Nikolov[5], Dominique Ritze[4], François Scharffe[6], Pavel Shvaiko[7], Ondřej Šváb-Zamazal[8], Cássia Trojahn[9], Ernesto Jimenez-Ruiz[11], Bernardo Cuenca Grau[11], and Benjamin Zapilko[10]

[1] INRIA & LIG, Montbonnot, France
{Jose-Luis.Aguirre,Jerome.Euzenat}@inria.fr
[2] Università degli studi di Milano, Italy
alfio.ferrara@unimi.it
[3] Vrije Universiteit Amsterdam, The Netherlands
{W.R.van.Hage,L.Hollink}@vu.nl
[4] University of Mannheim, Mannheim, Germany
{christian,dominique,kai}@informatik.uni-mannheim.de
[5] The Open University, Milton Keynes, United Kingdom
A.Nikolov@open.ac.uk
[6] LIRMM, Montpellier, France
francois.scharffe@lirmm.fr
[7] TasLab, Informatica Trentina, Trento, Italy
pavel.shvaiko@infotn.it
[8] University of Economics, Prague, Czech Republic
ondrej.zamazal@vse.cz
[9] IRIT – Université Toulouse II, Toulouse, France
cassia.trojahn@irit.fr
[10] GESIS – Leibniz Institute for the Social Sciences, Cologne, Germany
benjamin.zapilko@gesis.org
[11] University of Oxford, UK
{ernesto,berg}@cs.ox.ac.uk

Abstract. Ontology matching consists of finding correspondences between semantically related entities of two ontologies. OAEI campaigns aim at comparing ontology matching systems on precisely defined test cases. These test cases can use ontologies of different nature (from simple thesauri to expressive OWL ontologies) and use different modalities, e.g., blind evaluation, open evaluation, consensus. OAEI 2012 offered 7 tracks with 9 test cases followed by 21 participants. Since 2010, the campaign has been using a new evaluation modality which provides more automation to the evaluation. This paper is an overall presentation of the OAEI 2012 campaign.

* This paper improves on the "Preliminary results" initially published in the on-site proceedings of the ISWC workshop on Ontology Matching (OM-2012). The only official results of the campaign, however, are on the OAEI web site.

1 Introduction

The Ontology Alignment Evaluation Initiative[1] (OAEI) is a coordinated international initiative, which organizes the evaluation of the increasing number of ontology matching systems [12; 10; 26]. The main goal of OAEI is to compare systems and algorithms on the same basis and to allow anyone for drawing conclusions about the best matching strategies. Our ambition is that, from such evaluations, tool developers can improve their systems.

Two first events were organized in 2004: (i) the Information Interpretation and Integration Conference (I3CON) held at the NIST Performance Metrics for Intelligent Systems (PerMIS) workshop and (ii) the Ontology Alignment Contest held at the Evaluation of Ontology-based Tools (EON) workshop of the annual International Semantic Web Conference (ISWC) [27]. Then, a unique OAEI campaign occurred in 2005 at the workshop on Integrating Ontologies held in conjunction with the International Conference on Knowledge Capture (K-Cap) [1]. Starting from 2006 through 2011 the OAEI campaigns were held at the Ontology Matching workshops collocated with ISWC [11; 9; 3; 6; 7; 8]. In 2012, the OAEI results will be presented again at the Ontology Matching workshop[2] collocated with ISWC, in Boston, USA.

Since last year, we have been promoting an environment for automatically processing evaluations (§2.2), which has been developed within the SEALS (Semantic Evaluation At Large Scale) project[3]. SEALS provided a software infrastructure, for automatically executing evaluations, and evaluation campaigns for typical semantic web tools, including ontology matching. An intermediate campaign was executed in March 2012 in coordination with the Second SEALS evaluation campaigns. This campaign, called OAEI 2011.5, had five tracks and 18 participants, and it only ran on the SEALS platform. The results of OAEI 2011.5 have been independently published on the OAEI web site and are now integrated in this paper with those of OAEI 2012. For OAEI 2012, almost all of the OAEI data sets were evaluated under the SEALS modality, providing a more uniform evaluation setting.

This paper synthetizes the 2012 evaluation campaign and introduces the results provided in the papers of the participants. The remainder of the paper is organized as follows. In Section 2, we present the overall evaluation methodology that has been used. Sections 3-9 discuss the settings and the results of each of the test cases. Section 10 overviews lessons learned from the campaign. Finally, Section 11 concludes the paper.

2 General methodology

We first present the test cases proposed this year to the OAEI participants (§2.1). Then, we discuss the resources used by participants to test their systems and the execution environment used for running the tools (§2.2). Next, we describe the steps of the OAEI campaign (§2.3-2.5) and report on the general execution of the campaign (§2.6).

[1] http://oaei.ontologymatching.org
[2] http://om2012.ontologymatching.org
[3] http://www.seals-project.eu

2.1 Tracks and test cases

This year's campaign consisted of 7 tracks gathering 9 data sets and different evaluation modalities:

The benchmark track (§3): Like in previous campaigns, a systematic benchmark series has been proposed. The goal of this benchmark series is to identify the areas in which each matching algorithm is strong or weak by systematically altering an ontology. This year, like in OAEI 2011, we used new systematically generated benchmarks, based on four ontologies other than the original bibliographic one. For three of these ontologies, the evaluation was performed in blind mode.

The expressive ontologies track offers real world ontologies using OWL modelling capabilities:

Anatomy (§4): The anatomy real world case is about matching the Adult Mouse Anatomy (2744 classes) and a part of the NCI Thesaurus (3304 classes) describing the human anatomy.

Conference (§5): The goal of the conference task is to find all correct correspondences within a collection of ontologies describing the domain of organizing conferences (the domain being well understandable for every researcher). Results were evaluated automatically against reference alignments and by using logical reasoning techniques.

Large biomedical ontologies (§8): This track aims at finding alignments between large and semantically rich biomedical ontologies such as FMA, SNOMED CT, and NCI. The UMLS Metathesaurus has been selected as the basis for the track's reference alignments.

Multilingual

Multifarm(§6): This dataset is composed of a subset of the Conference dataset, translated in eight different languages (Chinese, Czech, Dutch, French, German, Portuguese, Russian, and Spanish) and the corresponding alignments between these ontologies.

Directories and thesauri

Library(§7): The library track is a real-word task to match two thesauri. The goal of this track is to find whether the matchers can handle such lightweight ontologies including a huge amount of concepts and additional descriptions. Results are evaluated both against a reference alignment and through manual scrutiny.

Instance matching (§9): The goal of the instance matching track is to evaluate the performance of different tools on the task of matching RDF individuals which originate from different sources but describe the same real-world entity. Instance matching is organized in two sub-tasks:

Sandbox: The Sandbox is a simple dataset that has been specifically conceived to provide examples of some specific matching problems (like name spelling and other controlled variations). This is intended to serve as a test for those tools that are in an initial phase of their development process and/or for tools that are facing very focused tasks, such as person name matching.

IIMB: IIMB is an OWL-based dataset that is automatically generated by introducing a set of controlled transformations in an initial OWL Abox, in order: i) to provide an evaluation dataset for various kinds of data transformations, including value transformations, structural transformations, and logical transformations; ii) to cover a wide spectrum of possible techniques and tools.

Table 1 summarizes the variation in the tests under consideration.

test	formalism	relations	confidence	modalities	language	SEALS
benchmark	OWL	=	[0 1]	blind+open	EN	√
anatomy	OWL	=	[0 1]	open	EN	√
conference	OWL-DL	=, <=	[0 1]	blind+open	EN	√
multifarm	OWL	=	[0 1]	open	CZ, CN, DE, EN, ES, DE, FR, RU, PT	√
library	OWL	=	[0 1]	open	EN, DE	√
large bio	OWL	=	[0 1]	open	EN	√
sandbox	RDF	=	[0 1]	open	EN	
iimb	RDF	=	[0 1]	open	EN	

Table 1. Characteristics of the test cases (open evaluation is made with already published reference alignments and blind evaluation is made by organizers from reference alignments unknown to the participants).

We do not present the New York Times (NYT) sub-track, held in the instance matching track, since it had only one participant.

2.2 The SEALS platform

In 2010, participants of the Benchmark, Anatomy and Conference tracks were asked for the first time to use the SEALS evaluation services: they had to wrap their tools as web services and the tools were executed on the machines of the tool developers [28]. Since 2011, tool developers had to implement a simple interface and to wrap their tools in a predefined way including all required libraries and resources. A tutorial for tool wrapping was provided to the participants. This tutorial describes how to wrap a tool and how to use a simple client to run a full evaluation locally. After local tests are passed successfully, the wrapped tool was uploaded for a test on the SEALS portal[4]. Consequently, the evaluation was executed by the organizers with the help of the SEALS technology. This approach allowed to measure runtime and ensured the reproducibility of the results. As a side effect, this approach ensures also that a tool is executed with the same settings for all of the six tracks that were executed in the SEALS mode. This was already requested in the previous years, however, this rule was sometimes ignored by participants.

2.3 Preparatory phase

Ontologies to be matched and (where applicable) reference alignments have been provided in advance during the period between June 15^{th} and July 1^{st}, 2012. This gave

[4] http://www.seals-project.eu/join-the-community/

potential participants the occasion to send observations, bug corrections, remarks and other test cases to the organizers. The goal of this preparatory period is to ensure that the delivered tests make sense to the participants. The final test base was released on July 6^{th}, 2012. The data sets did not evolve after that.

2.4 Execution phase

During the execution phase, participants used their systems to automatically match the test case ontologies. In most cases, ontologies are described in OWL-DL and serialized in the RDF/XML format [4]. Participants can self-evaluate their results either by comparing their output with reference alignments or by using the SEALS client to compute precision and recall. They can tune their systems with respect to the non blind evaluation as long as the rules published on the OAEI web site are satisfied. This phase has been conducted between July 6^{th} and August 31^{st}, 2012.

2.5 Evaluation phase

Participants have been encouraged to provide (preliminary) results or to upload their wrapped tools on the SEALS portal by September 1^{st}, 2012. For the SEALS modality, a full-fledged test including all submitted tools has been conducted by the organizers and minor problems were reported to some tool developers, until finally a properly executable version of all the tools has been uploaded on the SEALS portal.

First results were available by September 22^{nd}, 2012. The track organizers provided these results individually to the participants. The results were published on the respective web pages by the track organizers by October 15^{th}. The standard evaluation measures are precision and recall computed against the reference alignments. For the matter of aggregation of the measures, we used weighted harmonic means (weights being the size of the true positives). Another technique that was used is the computation of precision/recall graphs so it was advised that participants provide their results with a weight to each correspondence they found. We also computed for some tracks the degree of alignment coherency. Additionally, we measured runtimes for all tracks conducted under the SEALS modality.

2.6 Comments on the execution

For a few years, the number of participating systems has remained roughly stable: 4 participants in 2004, 7 in 2005, 10 in 2006, 17 in 2007, 13 in 2008, 16 in 2009, 15 in 2010, 18 in 2011, 21 in 2012. However, participating systems are now constantly changing. In 2012, 7 systems have not participated in any of the previous campaigns. The list of participants is summarized in Table 2.

This year only four systems participated in the instance matching track; two of them (LogMap and LogMapLt) participated also in the SEALS tracks.

Two tools, OMR and OntoK, are not shown in the table and were not included in the final evaluation.

OMR generated alignments with correspondences containing non-existing entities in one or both of the ontologies being matched. Moreover, it seems that its alignments

System	Aroma	ASE	AUTOMSv2	CODI	GOMMA	Hertuda	Hotmatch	LogMap	LogMapLt	MaasMtch	MapSSS	MEDLEY	Optima	SBUEI	semsim	ServOMap	ServOMapLt	TOAST	WeSeE	WikiMatch	YAM++	Total=21
Confidence	√	√	√		√	√		√		√		√				√	√	√	√	√	√	14
benchmarks	√	√	√		√	√	√	√	√	√	√	√	√			√	√		√	√	√	17
anatomy	√			√	√	√	√	√	√	√	√		√			√	√	√	√	√	√	16
conference	√	√	√	√	√	√	√	√	√	√	√	√	√			√	√		√	√	√	18
multifarm	√	√	√	√	√	√	√	√	√	√	√	√	√			√	√		√	√	√	18
library	√			√	√	√	√	√	√		√		√			√	√		√		√	13
large bio	√		√		√	√	√	√	√	√	√					√	√			√	√	13
sandbox								√	√					√								3
iimb								√	√					√	√							4
total	6	3	4	4	6	6	6	8	8	5	6	3	5	2	1	6	6	1	5	5	6	102

Table 2. Participants and the state of their submissions. Confidence stands for the type of result returned by a system: it is ticked when the confidence is a non boolean value.

are iteratively composed one by one, and that the last alignment contains all correspondences from other alignments but with changed namespaces, leading in many cases to very low precisions with quite high recalls. This behavior was reproduced along all the tracks.

OntoK revealed several bugs when preliminary tests were executed; the developers were not able to fix all of them before proceeding to the final evaluation.

Another tool was disqualified because we found that in quite a large number of cases across different tracks, the results it provided were too often exactly those of another matcher, including syntactic errors. After further scrutiny, it became obvious that the system implemented specific tricks to run well the OAEI tests instead of being a genuine matcher.

Finally, some systems were not able to pass some test cases as indicated in Table 2

The summary of the results track by track is presented in the following sections.

3 Benchmark

The goal of the benchmark data set is to provide a stable and detailed picture of each algorithm. For that purpose, algorithms are run on systematically generated test cases.

3.1 Test data

The systematic benchmark test set is built around a seed ontology and many variations of it. Variations are artificially generated, and focus on the characterization of the behavior of the tools rather than having them compete on real-life problems. They are organized in three groups:

Simple tests (1xx) such as comparing the reference ontology with itself;

Systematic tests (2xx) obtained by discarding/modifying features from the reference ontology. Considered features are names of entities, comments, the specialization hierarchy, instances, properties and classes.

Real-life ontologies (3xx) found on the web.

Full description of the systematic benchmark test set can be found on the OAEI web site.

This year the focus was on scalability, i.e., the ability of matchers to deal with data sets of increasing number of elements. To that extent, we departed from the usual bibliographic benchmark that has been used since 2004. We used a test generator [25] in order to reproduce the structure of benchmark for different seed ontologies, from different domains and with different sizes. We have generated five different benchmarks against which matchers have been evaluated:

benchmark (biblio) allows for comparison with other systems since 2004. The seed ontology concerns bibliographic references and is inspired freely from BibTeX. It contains 33 named classes, 24 object properties, 40 data properties, 56 named individuals and 20 anonymous individuals. This year we have used a new automatically generated version for this benchmark.

benchmark2 is related with the commerce domain. Its seed ontology contains 74 classes, 106 object properties and 35 named individuals.

benchmark3 is related with bioinformatics. Its seed ontology contains 233 classes, 83 object properties, 38 data properties and 681 named individuals.

benchmark4 is related with the product design domain. Its seed ontology contains 182 classes, 88 object properties, 202 data properties and 376 named individuals.

benchmark5 (finance) is based on the Finance ontology[5], which contains 322 classes, 247 object properties, 64 data properties and 1113 named individuals. It has been already considered in OAEI 2011 and 2011.5.

Having these five data sets also allowed us to better evaluate the dependency between the results and the seed ontology. **biblio** and **finance** were disclosed to the participants; the other benchmarks were tested in blind mode.

For all data sets, the reference alignments are still limited: they only match named classes and properties and use the "=" relation with confidence of 1.

3.2 Results

We evaluated 18 systems from the 22 participating in the SEALS tracks (Table 2). Besides OMR, OntoK and TOAST, excluded for reasons already explained, requirements for executing CODI in our machines were not met due to software license problems. In the following, we present the evaluation results.

[5] `http://www.fadyart.com/ontologies/data/Finance.owl`

Compliance Benchmark compliance tests have been executed on two cores and 8GB RAM Debian virtual machines (VM) running continuously in parallel, except for the finance data set which required 10GB RAM for some systems. For each benchmark seed ontology, data sets of 94 tests were automatically generated. We excluded from the whole systematic benchmark test set (111 tests), the tests that were not automatically generated: 102–104, 203–210, 230–231, 301–304.

Table 3 shows the compliance results (harmonic means of precision, F-measure and recall) of the five benchmark data sets for all the participants, as well as those given by edna, a simple edit distance algorithm on labels which is used as a baseline. The table also presents the confidence-weighted values of the same parameters.

Only ASE presented problems to process the **finance** data set, and MEDLEY did not completed the evaluation of the **benchmark4** and the **finance** data sets in a reasonable amount of time (12 hours).

Table 3 shows that, with few exceptions, all systems achieve higher levels of precision than recall for all benchmarks. Besides, no tool had a worst precision performance than the baseline, and only ServOMapLt had a significantly lower recall, with LogMap having slightly lower values for the same measure.

For those systems which have provided their results with confidence measures different from 1 or 0 (see Table 2), it is possible to draw precision/recall graphs and to compute weighted precision and recall. Systems providing accurate confidence values are rewarded by these measures [7]. Precision is increased for systems with many incorrect correspondences and low confidence, like edna and MaasMatch. Recall is decreased for systems with apparently many correct correspondences and low confidence, like AROMA, LogMap and YAM++. The variation for YAM++ is quite impressive, especially for the **biblio** benchmark.

Precision/recall graphs are given in Figure 1. The graphs show the real precision at n% recall and they stop when no more correspondences are available; then the end point corresponds to the precision and recall reported in Table 3.

Comparison across data sets From the results in Table 3, we observe that on average, all matchers have better performance than the baseline. The group of best systems in each data set remains relatively the same across the different benchmarks: YAM++, MapSSS and AROMA seems to generate the best alignments in terms of F-measure.

We also observe a high variance in the results of some systems across different benchmarks. Outliers are, for example, a poor precision for AROMA with **benchmark3** and a poor recall for ServOMapLt with **biblio**. These variations suggest interdependencies between matching systems and datasets that would need additional analysis requiring a deep knowledge of the evaluated systems. Such information is, in particular, useful for developers to detect and fix problems specific to their tools.

We also compare the results obtained by the tools that have participated in OAEI 2011 and 2012 on **biblio** and **finance** benchmarks which have been used in both campaigns. With respect to **biblio**, we observe negative variations between 2-4% for some tools, as well as positive variations between 1-3% for others. Regarding **finance**, the number of systems able to pass the tests increased, and for many tools that passed the tests in previous campaigns, positive variations between 1-3% were observed.

system	**refalign**			**edna**			**AROMA**			**ASE**			**AUTOMSv2**			**GOMMA**			**Hertuda**		
test	P	F	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F	R
biblio	1.0	1.0	1.0	.35(.45)	.41(.47)	.50	.98(.99)	.77(.73)	.64(.58)	.49	.51(.52)	.54	.97	.69	.54	.75(.74)	.67(.65)	.61(.58)	.90	.68	.54
benchmark2	1.0	1.0	1.0	.46(.61)	.48(.55)	.50	.97(.98)	.76(.73)	.63(.58)	.72(.74)	.61	.53	.97	.68	.52	.97	.69(.67)	.53(.51)	.93	.67	.53
benchmark3	1.0	1.0	1.0	.22(.25)	.30(.33)	.50	.38(.43)	.53(.54)	.83(.73)	.27	.36	.54	.99(1.0)	.70	.54	.99(1.0)	.70(.69)	.54(.52)	.94	.68	.54
benchmark4	1.0	1.0	1.0	.31(.37)	.38(.42)	.50	.96	.73(.70)	.59(.55)	.40(.41)	.45	.51	.91(.92)	.65	.51(.50)	.86	.63(.62)	.50(.49)	.90	.66	.51
finance	1.0	1.0	1.0	.22(.25)	.30(.33)	.50	.94	.72(.70)	.58(.56)	n.a.	n.a.	n.a.	.35	.42(.39)	.55(.46)	.95(.94)	.66(.64)	.51(.49)	.72	.62	.55

	HotMatch			**LogMap**			**LogMapLt**			**MaasMatch**			**MapSSS**			**MEDLEY**			
test	P	F	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F	R	
biblio	.96	.66	.50	.73	.56(.51)	.45(.39)	.71	.59	.50	.54(.90)	.56(.63)	.57(.49)	.99	.87	.77	.60(.59)	.54(.53)	.50 (.48)	
benchmark2	.99	.68	.52	1.0	.64(.59)	.47(.42)	.95	.66	.50	.60(.93)	.60(.65)	.60(.50)	1.0	.86	.76(.75)	.92(.94)	.65(.63)	.50 (.48)	
benchmark3	.99	.68	.52	.95(.96)	.65(.60)	.49(.44)	.95	.65	.50	.53(.90)	.53(.63)	.53(.48)	1.0	.82	.70	.78	.61(.56)	.50 (.43)	
benchmark4	.99	.66	.50	.99(1.0)	.63(.58)	.46(.41)	.95	.65	.50	.54(.92)	.54(.64)	.54(.49)	1.0(.99)	.81	.68	to	to	to	
finance	.97	.67	.51	.95(.94)	.63(.57)	.47(.40)	.90	.66	.52	.59(.92)	.59(.62)	.60(.48)	.99	.83	.71	to	to	to	

	Optima			**ServOMap**			**ServOMapLt**			**WeSeE**			**WikiMatch**			**YAM++**			
test	P	F	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F	R	
biblio	.89	.63	.49	.88	.58	.43	1.0	.33	.20	.99	.69(.68)	.53(.52)	.74	.62	.54	.98(.95)	.83(.18)	.72(.10)	
benchmark2	1.0	.66	.50	1.0	.67	.50	1.0	.51	.35(.34)	1.0	.69(.68)	.52	.97	.67(.68)	.52	.96(1.0)	.89(.72)	.82(.56)	
benchmark3	.97	.69	.53	1.0	.67	.50	1.0	.55	.38	1.0	.70(.69)	.53	.96(.97)	.68	.52	.97(1.0)	.85(.70)	.76(.54)	
benchmark4	.92	.60	.45	.89	.60(.59)	.45(.44)	1.0	.41	.26	1.0	.67(.66)	.50	.94(.95)	.66	.51	.96(1.0)	.83(.70)	.72(.54)	
finance	.96	.66	.51	.92	.63	.48	.99	.51(.50)	.34	.99	.70(.69)	.54(.53)	.74(.75)	.62(.63)	.54	.97(1.0)	.90(.72)	.84(.57)	

Table 3. Results obtained by participants on the five benchmark test cases aggregated with harmonic means (values within parentheses are *weighted* version of the measure reported only when different).

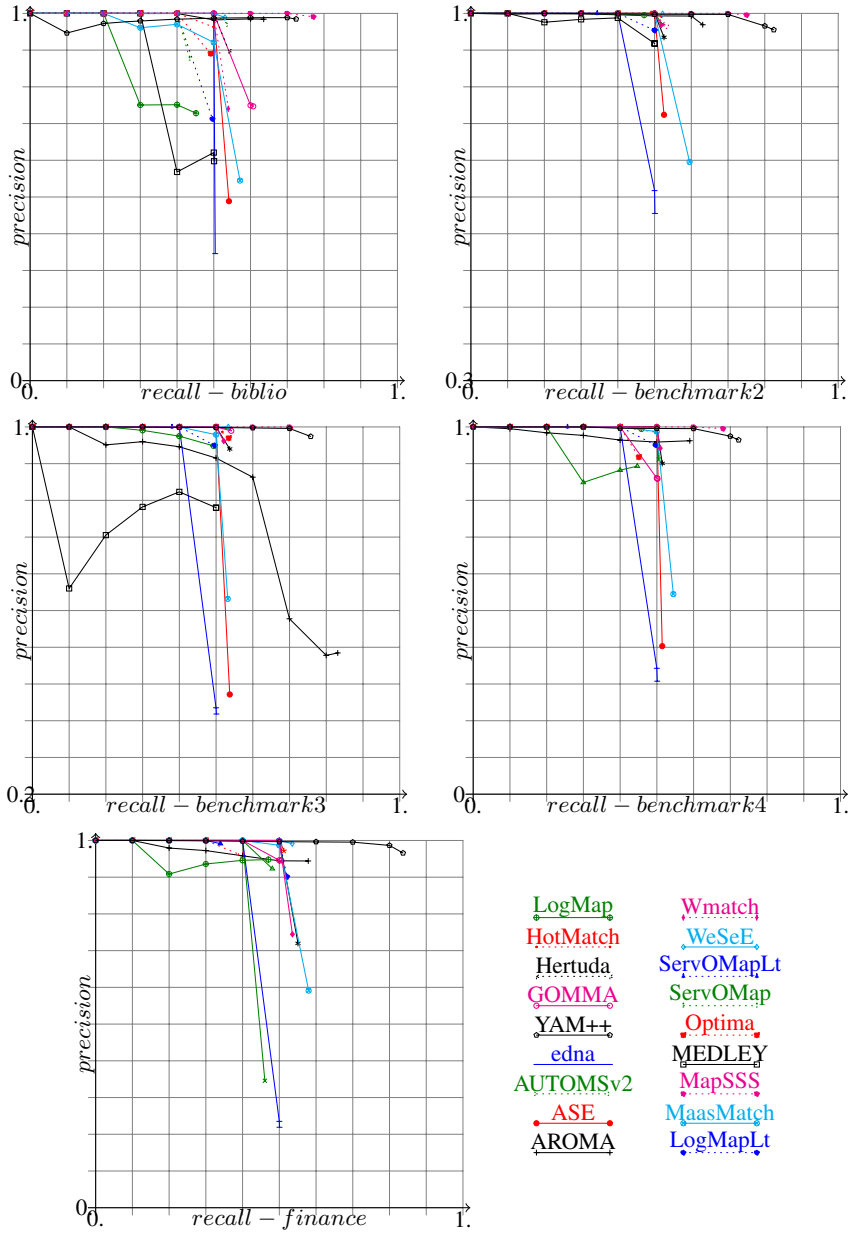


Fig. 1. Precision/recall graphs for benchmarks. The alignments generated by matchers are cut under a threshold necessary for achieving $n\%$ recall and the corresponding precision is computed. Systems for which these graphs are not meaningful (because they did not provide graded confidence values) are drawn in dashed lines.

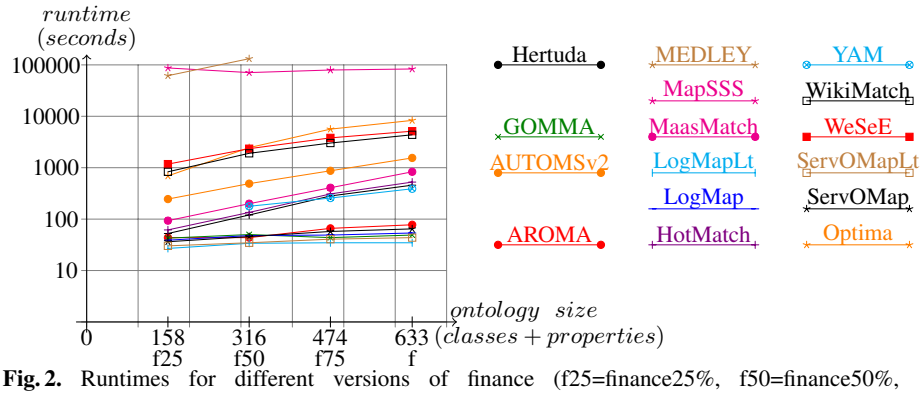


Fig. 2. Runtimes for different versions of finance (f25=finance25%, f50=finance50%, f75=finance75%, f=finance).

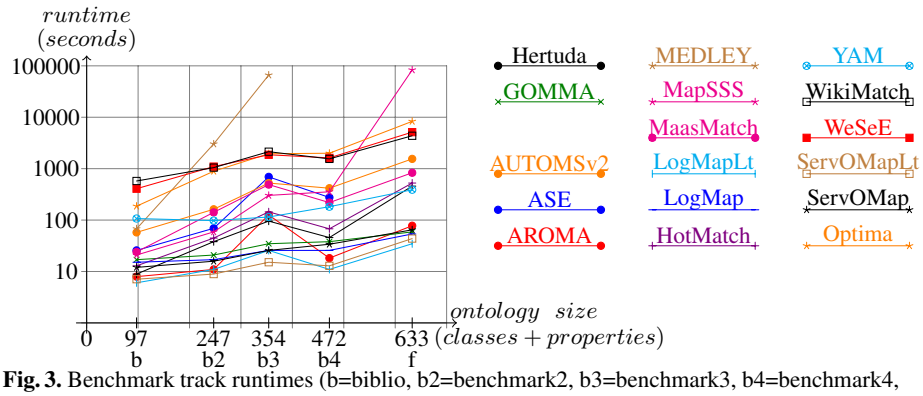


Fig. 3. Benchmark track runtimes (b=biblio, b2=benchmark2, b3=benchmark3, b4=benchmark4, f=finance).

Runtime Regarding runtime, scalability has been evaluated from two perspectives: on the one hand we considered the five seed ontologies from different domains and with different sizes; on the other hand we considered the **finance** ontology scaling it by reducing its size by different factors (25%, 50% and 75%). For the two modalities, the data sets were composed of a subset containing 15 tests extracted from a whole systematic benchmark.

All the experiments were done on a 3GHz Xeon 5472 (4 cores) machine running Linux Fedora 8 with 8GB RAM. Figures 2 and 3 show semi-log graphs for runtime measurements against data set sizes in terms of classes and properties.

First of all we observe that for the **finance** tests, the majority of tools have a monotonic increasing run time, with the exception of MapSSS which exhibits an almost constant response time. On the contrary, this does not happen for the benchmark tests, for which the **benchmark3** test causes a break in the monotonic behavior. One reason for this could be that the **benchmark3** ontology has a more complex structure than the other ones, and that matchers basing their work in structural analysis are more affected than others.

Figures also show that there is a set of tools that distance themselves from the others: LogMapLt, ServOMapLt, LogMap, ServOMap, GOMMA and Aroma are the fastest tools, and are able to process large ontologies in a short time. On the contrary, there exist tools that were not able to deal with large ontologies in the same conditions: MEDLEY and MapSSS fall in this category.

3.3 Conclusions

Having five different benchmarks allowed us to see the degree of dependency on the shapes and sizes of the seed ontologies. Even if differences were observed in the results obtained, we can conclude that excepting a few cases, the tools do not show large variations in the results to the different benchmarks.

Regarding compliance, we observed that with very few exceptions, the systems performed always better than the baseline. However, there were no significant improvements in the performance of the systems with respect to their performance in last OAEI campaigns (OAEI 2011 and 2011.5).

Regarding runtime, we noticed that given ontologies of different sizes sharing the structure and knowledge domain to a big extent, the response time follows generally the shape of a monotonic increasing function. On the contrary, this is not always true if the shapes or the knowledge domains of the ontologies change.

The results obtained this year allow us to confirm that we cannot conclude on a general correlation between runtime and quality of alignments. The slowest tools do not necessarily provide the best compliance results.

4 Anatomy

The anatomy track confronts matchers with a specific type of ontologies from the biomedical domain. In this domain, many ontologies have been built covering different aspects of medical research. We focus on two fragments of biomedical ontologies which describe the human anatomy and the anatomy of the mouse. The data set of this track has been used since 2007 with some improvements over the last years. For a detailed description, we refer the reader to the OAEI 2007 results paper [9].

4.1 Experimental setting

Contrary to previous years, we conducted only a single evaluation experiment by executing each matcher in its standard setting. In our experiments, we compare precision, recall, F-measure and recall+. The measure recall+ indicates the amount of detected non-trivial correspondences. The matched entities in a non-trivial correspondence do not have the same normalized label. The approach that generates only trivial correspondences is depicted as baseline *StringEquiv* in the following section. In OAEI 2011/2011.5, we executed the systems on our own (instead of analyzing submitted alignments) and reported about measured runtimes. Unfortunately, we did not use exactly the same machine compared to previous years. Thus, runtime results are not fully comparable across years. In 2012, we used an Ubuntu machine with 2.4 GHz (2

cores) and 3GB RAM allocated to the matching systems. Further, we used the SEALS client to execute our evaluation. However, we slightly changed the way precision and recall are computed, i.e., the results generated by the SEALS client vary in some cases by 0.5% compared to the results presented below. In particular, we remove trivial correspondences in the `oboInOwl` namespace like

```
http://...oboInOwl#Synonym = http://...oboInOwl#Synonym
```

as well as correspondences expressing relations different from equivalence. We also checked whether the generated alignment is coherent, i.e., there are no unsatisfiable concepts when the ontologies are merged with the alignment.

4.2 Results

In Table 4, we listed all the participating systems that generated an alignment in less than ten hours. The listing comprises 17 entries. Three systems participated each with two different versions. This is GOMMA (the extension "'-bk"' refers to the usage of background knowledge), LogMap and ServoMap (both systems have submitted an additional lightweight version that uses only some core components). Thus, 14 different systems generated an alignment within the given time frame. There were three participants ASE, AUTOMSv2, and MEDLEY that did no finish in time or threw an exception. Due to several hardware and software requirements, we could not install TOAST on the machine on which we executed the other systems. We executed the matcher on a different machine of similar strength. For this reason, the runtime of TOAST is not fully comparable to the other runtimes (indicated by an asterisk).

Compared to previous years, we can observe a clear speed increase. In 2012, five systems (counting two versions of the same system as one) finished in less than 100 seconds, compared to two systems in OAEI 2011 and three systems in OAEI 2011.5. This has to be mentioned as a positive trend. Moreover, in 2012 we were finally able to generate results for 14 of 17 systems, while in 2011 only 7 of 14 systems generated results of acceptable quality within the given time frame. The top systems in terms of runtimes are GOMMA, LogMap and ServOMap. Depending on the specific version of the systems, they require between 6 and 34 seconds to match the ontologies. Table 4 shows that there is no correlation between the quality of the generated alignment in terms of precision and recall and the required runtime. This result has also been observed in previous campaigns.

Table 4 also shows the results for precision, recall and F-measure. We ordered the matching systems with respect to the achieved F-measure. The F-measure is an aggregation of precision and recall. Depending on the application for which the generated alignment is used, it might, for example, be more important to favor precision over recall or vice versa. In terms of F-measure, GOMMA-bk is ahead of the other participants. The differences of GOMMA-bk compared to GOMMA (and the other systems) are based on mapping composition techniques and the reuse of mappings between UMLS, Uberon and FMA. GOMMA-bk is followed by a group of matching systems (YAM++, CODI, LogMap, GOMMA) generating alignments that are very similar with respect to precision, recall and F-measure (between 0.87 and 0.9 F-measure). To our knowledge,

Matcher	Runtime(s)	Size	Precision	F-measure	Recall	Recall+	Coherent
GOMMA-bk	15	1534	0.917	0.923	0.928	0.813	-
YAM++	69	1378	0.943	0.898	0.858	0.635	-
CODI	880	1297	0.966	0.891	0.827	0.562	√
LogMap	20	1392	0.920	0.881	0.845	0.593	√
GOMMA	17	1264	0.956	0.870	0.797	0.471	-
MapSSS	453	1212	0.935	0.831	0.747	0.337	-
WeSeE	15833	1266	0.911	0.829	0.761	0.379	-
LogMapLt	6	1147	0.963	0.829	0.728	0.290	-
TOAST	3464*	1339	0.854	0.801	0.755	0.401	-
ServOMap	34	972	0.996	0.778	0.639	0.054	-
ServOMapL	23	976	0.990	0.775	0.637	0.052	-
HotMatch	672	989	0.979	0.773	0.639	0.145	-
AROMA	29	1205	0.865	0.766	0.687	0.321	-
StringEquiv	-	946	0.997	0.766	0.622	0.000	-
Wmatch	17130	1184	0.864	0.758	0.675	0.157	-
Optima	6460	1038	0.854	0.694	0.584	0.133	-
Hertuda	317	1479	0.690	0.681	0.673	0.154	-
MaasMatch	28890	2737	0.434	0.559	0.784	0.501	-

Table 4. Comparison against the reference alignment, runtime is measured in seconds, the "size" column refers to the number of correspondences in the generated alignment.

these systems either do not use specific background knowledge for the biomedical domain or use it only in a very limited way. The results of these systems are at least as good as the results of the best system in OAEI 2007-2010, only AgreementMaker, using additional background knowledge, could generate better results in OAEI 2011. Most of the evaluated systems achieve an F-measure that is higher than the baseline that is based on (normalized) string equivalence. Moreover, nearly all systems find many non-trivial correspondences. An exception is the system ServOMap (and its lightweight version) that generates an alignment that is quite similar to the alignment generated by the baseline approach.

Concerning alignment coherency, only CODI and LogMap generated coherent alignments. We have to conclude that there have been no improvements compared to OAEI 2011 with respect to taking alignment coherence into account. LogMap and CODI generated a coherent alignment already in 2011. Furthermore, is can be observed (see Section 5) that YAM++ generates coherent alignments for the ontologies of the Conference track, which are much smaller but more expressive, while it fails to generate coherent alignments for larger biomedical ontologies (see also Section 8). This might be based on using different settings for larger ontologies to avoid reasoning problems with larger input.

4.3 Conclusions

Most of the systems top the string equivalence baseline with respect to F-measure. Moreover, we reported that several systems achieve very good results compared to the

evaluations of the previous years. A clear improvement compared to previous years can be seen in the number of systems that are able to generate such results. It is also a positive trend that more matching systems can create good results within short runtimes. This might partially be caused by offering the Anatomy track constantly in its current form over the last six years together with publishing matcher runtimes. At the same time, new tracks that deal with large (and very large) matching tasks are offered. These tasks can only be solved with efficient matching strategies that have been implemented over the last years.

5 Conference

The conference test case introduces matching several moderately expressive ontologies. Within this track, participant results were evaluated against reference alignments (containing merely equivalence correspondences) and by using logical reasoning. As last year, the evaluation has been supported by the SEALS technology. This year we used refined and harmonized reference alignments.

5.1 Test data

The collection consists of sixteen ontologies in the domain of organizing conferences. These ontologies have been developed within the OntoFarm project[6].

The main features of this test case are:

- *Generally understandable domain.* Most ontology engineers are familiar with organizing conferences. Therefore, they can create their own ontologies as well as evaluate the alignments among their concepts with enough erudition.
- *Independence of ontologies.* Ontologies were developed independently and based on different resources, they thus capture the issues in organizing conferences from different points of view and with different terminologies.
- *Relative richness in axioms.* Most ontologies were equipped with OWL DL axioms of various kinds; this opens a way to use semantic matchers.

Ontologies differ in their numbers of classes, of properties, in expressivity, but also in underlying resources.

5.2 Results

This year, we provide results in terms of $F_{0.5}$-measure, F_1-measure and F_2-measure, comparison with baseline matcher, precision/recall triangular graph and coherency evaluation.

[6] http://nb.vse.cz/~svatek/ontofarm.html

Evaluation based on reference alignments We evaluated the results of participants against new reference alignments (labelled as *ra2* on the conference web-page). This includes all pairwise combinations between 7 different ontologies, i.e. 21 alignments.

New reference alignments have been generated as a transitive closure computed on the original reference alignments. In order to obtain a coherent result, conflicting correspondences, i.e., those causing unsatisfiability, have been manually inspected and removed. As a result the degree of correctness and completeness of the new reference alignment is probably slightly better than for the old one. However, the differences are relatively limited. Whereas the new reference alignments are not open, the old reference alignments (labeled as *ra1* on the conference web-page) are available. These represent close approximation of the new ones.

Matcher	Precision	$F_{0.5}$-measure	F_1-measure	F_2-measure	Recall
YAM++	.78	.75	.71	.67	.65
LogMap	.77	.71	.63	.57	.53
CODI	.74	.69	.63	.58	.55
Optima	.60	.61	.61	.62	.63
GOMMA	.79	.68	.56	.47	.43
Hertuda	.70	.63	.56	.49	.46
MaasMatch	.60	.58	.56	.53	.52
Wmatch	.70	.63	.55	.48	.45
WeSeE	.72	.64	.55	.48	.44
HotMatch	.67	.62	.55	.50	.47
LogMapLt	.68	.62	.54	.48	.45
Baseline	*.76*	*.64*	*.52*	*.43*	*.39*
ServOMap	.68	.60	.51	.45	.41
ServOMapLt	.82	.65	.50	.41	.36
MEDLEY	.59	.55	.49	.45	.42
ASE*	.61	.55	.48	.43	.40
MapSSS	.47	.47	.46	.46	.46
AUTOMSv2*	.64	.54	.44	.37	.33
AROMA	.33	.34	.37	.39	.41

Table 5. The highest average $F_{[0.5|1|2]}$-measure and their corresponding precision and recall for each matcher F_1-optimal threshold.

Table 5 shows the results of all participants with regard to the new reference alignment. $F_{0.5}$-measure, F_1-measure and F_2-measure are computed for the threshold that provides the highest average F_1-measure. F_1 is the harmonic mean of precision and recall where both are equally weighted; F_2 weights recall higher than precision and $F_{0.5}$ weights precision higher than recall. The matchers shown in the table are ordered according to their highest average F_1-measure. Our *baseline*[7] divides matchers into two groups. Group 1 consists of matchers (YAM++, LogMap, CODI, Optima, GOMMA, Hertuda, MaasMatch, Wmatch, WeSeE, HotMatch and LogMapLt) having better (or equal) results than *Baseline*. Other matchers (ServOMap, ServOMapLt, MEDLEY,

[7] String matcher based on string equality applied on local names of entities which were lowercased before.

ASE, MapSSS, AUTOMSv2 and AROMA) performed worse than *baseline*. There are two matchers (ASE and AUTOMSv2) with asterisks which did not generate 3 out of 21 alignments. Thus, their results are just an approximation.

Performance of matchers from Group 1 regarding F_1-measure is visualized in Figure 4.

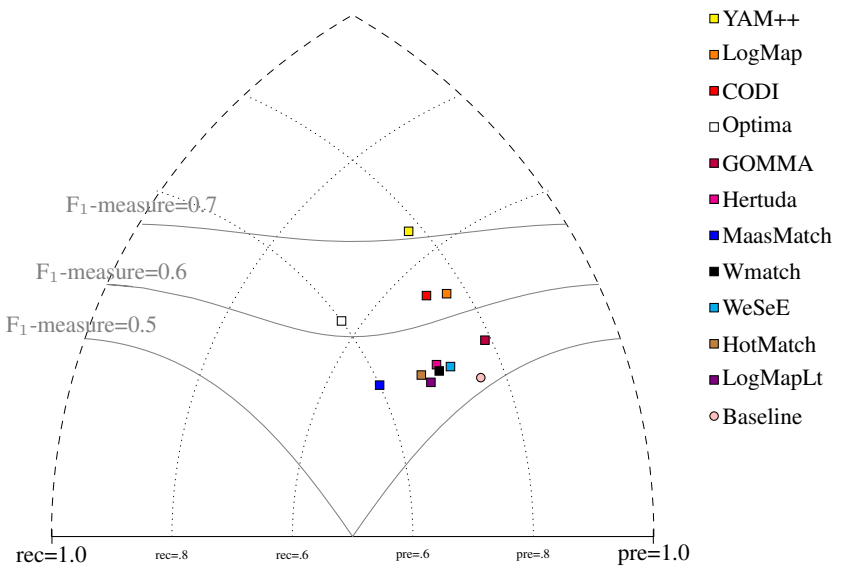


Fig. 4. Precision/recall triangular graph for the conference track. Matchers are represented as squares and Baseline is represented as a circle. Dotted lines depict level of precision/recall while values of F_1-measure are depicted by areas bordered by corresponding lines F_1-measure=0.[5|6|7].

Comparison with previous years Seven matchers also participated in OAEI 2011 and 10 matchers participated in OAEI 2011.5. The largest improvement was achieved by Optima (precision from .23 to .60 and recall from .52 to .63) and YAM++ (precision from .74 to .78 and recall from .51 to .65) between OAEI 2011 and 2012. Four matchers were improved between OAEI 2011.5 and 2012 and five matchers were improved between OAEI 2011 and 2012.

Runtimes We measured the total time of generating all 120 alignments. It was executed on a laptop with Ubuntu machine running on Intel Core i5, 2.67GHz and 4GB RAM. In all, four matchers finished all 120 test cases within 1 minute (LogMapLt - 44 seconds, Hertuda - 49 seconds, ServOMapLt - 50 seconds and AROMA - 55 seconds). Next, four matchers needed less than 2 minutes (ServOMap, HotMatch, GOMMA and ASE). 10 minutes were enough for the next four matchers (LogMap, MapSSS, MaasMatch and AUTOMSv2). Finally, 5 matchers needed up to 40 minutes to finish all 120 test cases (Optima - 22 min, MEDLEY - 30 min, WeSeE - 36 min, CODI - 39 min and Wmatch - 40 min). YAM++ did not finish the task of matching all 120 test cases within five hours.

In conclusion, regarding performance we can see (clearly from Figure 4) that YAM++ is on the top. Next three matchers (LogMap, CODI, and Optima) are relatively close to each other. This year there is a largest group of matchers which are above *baseline* than previous years. Moreover, it is very positive that several matchers managed to improve their performance in such a short time as one year or even half a year.

Evaluation based on alignment coherence As in previous years, we applied the Maximum Cardinality measure to evaluate the degree of alignment incoherence. Details on this measure and its implementation can be found in [19]. The results of our experiments are depicted in Table 6. Contrary to last year, we only compute the average for all test cases of the conference track for which there exists a reference alignment. The presented results are thus aggregated mean values for 21 test cases. In some cases we could not compute the degree of incoherence due to the combinatorial complexity of the problem. In this case we were still able to compute a lower bound for which we know that the actual degree is (probably only slightly) higher. Such results are marked with a *. Note that we only included in our evaluation those matchers that generated alignments for all test cases of the subset with reference alignments.

Matcher	AROMA	CODI	GOMMA	Hertuda	HotMatch	LogMap	LogMapLt	MaasMatch*
Alignment Size	20.9	11.1	8.2	9.9	10.5	10.3	9.9	83.1
Incoherence Degree	19.4%	0%	1.1%	5.2%	5%	0%	5.4%	24.3%
Incoherent Alignments	18	0	2	9	9	0	7	20

Matcher	MapSSS	MEDLEY*	Optima	ServOMap	ServOMapLt	WeSeE	Wmatch	YAM++
Alignment Size	14.8	55.6	15.9	9	6.5	9.4	9.9	12.5
Incoherence Degree	12.6%	30.5%	7.6%	3.3%	0%	3.2%	6%	0%
Incoherent Alignments	18	20	12	5	0	6	10	0

Table 6. Average size of alignments, average degree of incoherence, and number of incoherent alignments. The mark * is added if we only provide lower bound of the degree of incoherence due to the combinatorial complexity of the problem.

Four matchers can generate coherent alignments. These matchers are CODI, LogMap, ServOMapLt, and YAM++. However, it is not always clear whether this is related to a specific approach that tries to ensure the coherency, or whether this is only indirectly caused by generating small and highly precise alignments. In particular, the coherence of the alignments from ServOMapLt, which does not apply any semantic technique, might be caused by such an approach. The matcher generates overall the smallest alignments. Because there are some matchers that cannot generate a coherent alignment for alignments that have in average a size from 8 to 12 correspondences, it can

be assumed that CODI, LogMap, and YAM++ have implemented specific coherency-preserving methods. Those matchers generate also between 8 to 12 correspondences, however, none of their alignments is incoherent. This is an important improvement compared to the previous years, for which we observed that only one or two matchers managed to generate (nearly) coherent alignments.

6 MultiFarm

In order to be able to evaluate the ability of matching systems to deal with ontologies in different languages, the MultiFarm dataset has been proposed [21]. This dataset results from the translation of seven Conference track ontologies (cmt, conference, confOf, iasted, sigkdd, ekaw and edas), in eight languages: Chinese, Czech, Dutch, French, German, Portuguese, Russian, and Spanish (+ English). The translations in 8 languages + English result in 36 pairs of languages. Overall, we have 36×49 matching tasks (see [21] for details on the pairs of languages and ontologies).

6.1 Experimental setting

For the 2012 evaluation campaign, we have used a subset of the whole MultiFarm dataset, omitting all the pairs of matching tasks involving the ontologies edas and ekaw (resulting in 36×25 matching tasks). This allows for using the omitted test cases as blind evaluation tests in the future. Contrary to OAEI 2011.5, we have included the Chinese and Russian translations.

Within the MultiFarm dataset, we can distinguish two types of matching tasks: (i) those test cases in which two different ontologies have been translated in different languages (cmt–confOf, for instance); and (ii) those test cases where the same ontology has been translated in different languages (cmt–cmt, for instance). For the test cases of type (ii), good results are not directly related to the use of specific techniques for dealing with ontologies in different natural languages, but on the ability to exploit the fact that both ontologies have an identical structure (and that the reference alignment covers all entities described in the ontologies).

This year, seven participating systems (out of 21 systems participated in OAEI, see Table 2) use specific multilingual methods: ASE, AUTOMSv2, GOMMA, MEDLEY, WeSeE, Wmatch, and YAM++. The other systems are not specifically designed to match ontologies in different languages, nor do they make use of a component that can be used for that purpose.

ASE (a version of AUTOMSv2) uses the Microsoft Bing Translator API for translating the ontologies to English. This process is performed before ASE profiling, configuration and matching methods are executed, so its input will consider only English labeled copies of ontologies. AUTOMSv2 follows a similar approach, but re-using a free Java API named WebTranslator. GOMMA uses a free translation API (MyMemory), for translating non-English concept labels to English. The translations are associated to concepts as new synonyms. Iteratively, GOMMA creates a bilingual dictionary for each ontology, which is used within the matching process. WeSeE and YAM++, as AUTOMS2, use Microsoft Bing Translation for translating the labels contained in

the input ontologies to English. Then, the translated English ontologies are matched using standard matching procedures of WeSeE and YAM++. Finally, Wmatch exploits Wikipedia for extracting inter-language. All matchers (with the exception of Wmatch) use English as a pivot language. MEDLEY is the only matcher for which we have no information on the techniques it exploits to deal with multilingualism.

6.2 Execution setting and runtime

All systems (with the exception of CODI) have been executed on a 3GHz Xeon 5472 (4 cores) machine, running Linux Fedora 8 with 8GB RAM. The runtimes for each system can be found in Table 7. As CODI has been executed on a different setting, its runtime cannot be compared with the runtime of other systems. We observe large differences between the time required for a system to complete the 36×25 matching tasks. While WeSeE requires $\cong$ 15 minutes, Wmatch takes $\cong$ 17 hours. It is mainly due to the fact that Wmatch uses an external resource (Wikipedia) for looking for inter-languages links. This requires considerably more time than simpler requests for translations.

6.3 Overall results

Before discussing the results per pairs of languages, we present the aggregated results for the test cases within type (i) and (ii) matching task. Table 7 shows the aggregated results. Systems not listed in this table have generated empty alignments, for all test cases (ServOMap and ServOMapL), have thrown exceptions (ASE, OMR, OntoK), or have not been evaluated due to their execution requirements (TOAST). AROMA was not able to generate alignments for test cases of type (i).

As shown in Table 7, we can observe significant differences between the results obtained for each type of matching task, specially in terms of precision. While the systems that implement specific multilingual techniques clearly generate the best results for test cases of type (i), only one of these systems (YAM++) is among the top (3) F-measures for type (ii) test cases. For these test cases, MapSSS and CODI, which implement strategies to deal with ontologies that share structural similarities, have better results. Due to this feature, they have preserved their overall performance this year (using the same version as for the last campaign), even though harder tests have been included in 2012 (Chinese and Russian translations). On the other hand, for the other matchers in the same situation, the differences in the results are explained by the presence of such harder tests cases this year.

Furthermore, as observed in the OAEI 2011.5 campaign and corroborated in 2012, MapSSS and CODI have generated very good results on the benchmark track. This suggests a strong correlation between the ranking in Benchmark and the ranking for MultiFarm test cases of type (ii), while there is, on the other hand, no (or only a very weak) correlation between results for test cases of type (i) and type (ii). For that reason, we only analyze in the following the results for test cases of type (i).

			Different ontologies (i)			Same ontologies (ii)		
	System	Runtime	Prec.	Fmeas.	Rec.	Prec.	Fmeas.	Rec.
Multilingual	AUTOMSv2	512.7	**.49**	**.36**	.10	**.69**	.24	.06
	GOMMA	35.0	**.29**	**.31**	**.36**	.63	**.38**	.29
	MEDLEY	76.5	.16	**.16**	.07	.34	.18	.09
	WeSeE	14.7	**.61**	**.41**	**.32**	**.90**	**.41**	.27
	Wmatch	1072.0	.22	.21	**.22**	.43	.17	.11
	YAM++	367.1	**.50**	**.40**	**.36**	**.91**	**.60**	**.49**
Non specific	AROMA	6.9				.31	.01	.01
	CODI	x	**.17**	.08	.02	**.82**	**.62**	**.50**
	Hertuda	23.5	.00	.01	1.00	.02	.03	**1.00**
	HotMatch	16.5	.00	.01	.00	.40	.04	.02
	LogMap	14.9	**.17**	.09	.02	.35	.03	.01
	LogMapLt	5.5	.12	.07	.02	.30	.03	.01
	MaasMatch	125.0	.02	.03	.14	.14	.14	.14
	MapSSS	17.3	.08	.09	.04	**.97**	**.66**	**.50**
	Optima	142.5	.00	.01	**.59**	.02	.03	**.41**

Table 7. MultiFarm aggregated results per matcher, for each type of matching task – types (i) and (ii). Runtime is measured in minutes (time for completing the 36×25 matching tasks). The top-5 values for each column are marked in bold-face.

6.4 Language specific results

Table 8 shows the results aggregated per language pair. For the sake of readability, we present only F-measure values. The reader can refer to the OAEI results web page for more detailed results on precision and recall.

As expected and already reported above, the systems that apply specific strategies to deal with multilingual matching labels outperform all other systems (overall F-measure for both cases): YAM++, followed by WeSeE, GOMMA, AUTOMSv2, Wmatch, and MEDLEY, respectively. Wmatch has the ability to deal with all pairs of languages, what is not the case for AUTOMSv2 and MEDLEY, specially for the pairs involving Chinese, Czech and Russian languages.

Most of the systems translating non-English ontology labels to English have better scores on pairs where English is present (by group of pairs, YAM++ is the typical case). This owes to the fact that multiple translations (pt $\rightarrow$ en and fr $\rightarrow$ en, for matching pt $\rightarrow$ fr, for instance) may result in more ambiguous translated concepts, which makes harder the process of finding correct correspondences. Furthermore, as somehow expected, good results are also obtained for pairs of languages having similarities on their vocabularies (es-pt and fr-pt, for instance). These two observations may explain the top F-measures of the specific multilingual methods: AUTOMSv2 (es-pt, en-es, de-nl, en-nl), GOMMA (en-pt, es-pt, cn-en, de-en), MEDLEY (en-fr, en-pt, cz-en, en-es), WeseE (en-es, es-fr, en-pt, es-pt, fr-pt), YAM++ (cz-en, cz-pt, en-pt). Wmatch has an interesting pair score, where Russian appears in the top F-measures: nl-ru, en-es, en-nl, fr-ru, es-ru. This may be explained by the use of Wikipedia multilingual inter-links, which are not limited to English or language similarities.

Pair	AUTOMSv2	CODI	GOMMA	Herduda	HotMatch	LogMap	LogMapLt	MaasMtch	MapSSS	MEDLEY	Optima	WeSeE	Wmatch	YAM++
cn cz			.34	.01				.01			.01	.29	.21	.35
cn de			.34	.01				.01			.01	.25	.08	.35
cn en			.41	.01				.00			.01	.31	.10	.41
cn es			.33	.01	.01			.00			.01	.34	.14	.20
cn fr			.35	.01	.01			.00			.01	.32	.09	.39
cn nl			.25	.01	.01			.00			.01	.23	.10	.34
cn pt			.31	.01				.01			.01	.30	.09	.35
cn ru			.27	.01	.01			.00			.01	.27	.13	.33
cz de		.10	.24	.01		.10	.09	.06	.07	.19	.01	.41	.24	.45
cz en		.07	.36	.01		.05	.04	.06	.08	.28	.01	.48	.24	.58
cz es		.11	.30	.01		.11	.11	.06	.11	.13	.01	.47	.25	.20
cz fr		.01	.16	.01	.01	.01	.01	.04	.01	.08	.01	.47	.20	.53
cz nl		.09	.21	.01		.04	.04	.07	.05	.09	.01	.48	.21	.55
cz pt		.15	.37	.01		.13	.13	.06	.12	.18	.01	.44	.12	.57
cz ru			.21	.01				.00			.01		.17	.49
de en	.38	.20	.41	.01		.22	.20	.06	.16	.27	.01	.39	.28	.52
de es	.35	.06	.35	.01		.12	.06	.06	.15	.15	.01	.41	.24	.20
de fr	.32	.04	.21	.01		.04	.04	.05	.13	.13	.01	.41	.25	.46
de nl	.39	.05	.24	.01		.04	.04	.06	.15	.12	.01	.37	.25	.40
de pt	.35	.08	.36	.01		.07	.07	.05	.06	.15	.01	.35	.22	.42
de ru			.33	.01	.01			.01			.01	.42	.26	.47
en es	.42	.04	.40	.01		.15	.04	.08	.18	.28	.01	.52	.30	.23
en fr	.31	.04	.36	.01	.01	.06	.04	.09	.13	.33	.01	.48	.27	.53
en nl	.39	.10	.38	.01		.08	.10	.09	.15	.24	.01	.49	.29	.53
en pt	.37	.08	.45	.01		.06	.06	.06	.07	.30	.01	.51	.26	.56
en ru			.34	.01				.01			.01	.43	.25	.47
es fr	.37	.01	.29	.01		.07	.01	.08	.06	.06	.01	.52	.24	.20
es nl	.38		.33	.01				.05	.01	.03	.01	.46	.27	.16
es pt	.44	.22	.44	.01		.24	.23	.11	.23	.22	.01	.51	.25	.25
es ru			.21	.01				.00			.01		.28	.19
fr nl	.27	.13	.21	.01	.01	.13	.12	.07	.11	.16	.01	.43	.28	.47
fr pt	.35		.32	.01	.01			.06	.02	.08	.01	.50	.23	.53
fr ru			.24	.01				.00			.01	.47	.28	.46
nl pt	.37	.04	.32	.01		.01	.01	.05	.02	.06	.01	.47	.20	.51
nl ru			.22	.01				.01			.01	.42	.31	.42
pt ru			.30	.01				.00			.01	.45	.17	.44

Table 8. MultiFarm results per pair of language, for the test cases of type (i). We distinguished empty alignments, represented by empty cells, from wrong ones.

For non-specific systems, though all of them cannot deal at all with Chinese and Russian languages, MapSSS, LogMap and CODI obtain better results. These system perform better for some specific pairs: MapSSS (es-pt, en-es, de-en), LogMap and LogMapL (es-pt, de-en, en-es, cz-pt), CODI (es-pt, de-en, cz-pt). From all these sys-

tems, the pairs es-pt and de-en are obtain better F-measures. Again, we can see that similarities in the language vocabulary have an important role in the matching task. On the other hand, although it is likely harder to find correspondences between cz-pt than es-pt, for some systems their best score include such combinations (cz-pt, for CODI and LogMapLt). This can be explained by the specific way systems combine their internal matching techniques (ontology structure, reasoning, coherence, linguistic similarities, etc).

6.5 Conclusions

We observe that specific methods for dealing with ontologies that are described in different languages, work much better than non specific systems. This is the expected behavior. However, the absolute results are still not very good, if compared to the top results of the original Conference dataset ($\cong$ .75 F-measure for the best matcher). For all specific multilingual methods, the techniques implemented in YAM++ generate the best alignments in terms of F measure ($\cong$ .53 overall F-measure for both types of matching tasks). YAM++ is followed by WeSeE and GOMMA, respectively. With the exception of Wmatch, all systems use English as pivot language.

Looking at the participation in the OAEI 2011.5 campaign, only 3 participants, out of 19 have used specific techniques. We counted this year with new systems implementing specific multilingual methods (seven out of 21). Although there is room for improvements to achieve the same level of compliance than in the original dataset, the increasing number of matchers dealing with multilingual matching is a sign that the field is progressing.

7 Library

This library track is a new track within the OAEI. Its challenge is to match two real-world thesauri: TheSoz (social sciences) and STW (economics). However, there has already been a library track from 2007 to 2009 [9; 3; 6] using different thesauri,[8] as well as other thesaurus tracks like the food track[9] and the environment track.[10] A common motivation is that these tracks use a real-world scenario, i.e., real thesauri. For us, it is still a motivation to develop a better understanding, how thesauri differ from ontologies and how these differences affect state-of-the-art ontology matchers. We hope that the community accepts the challenge and that subsequently significant improvements can be seen that push the quality of automatic alignments between thesauri. Furthermore, we will use the matching results as input for the maintainers of the reference alignment to improve the alignment. While a full manual evaluation of all matching results is certainly not feasible, this way we constantly improve the reference alignment and mitigate possible weaknesses and incompleteness.

[8] http://oaei.ontologymatching.org/2009/library/
[9] http://oaei.ontologymatching.org/2006/food/
[10] http://oaei.ontologymatching.org/2007/environment/

7.1 Test data

The library track uses two real-world thesauri, that are in many aspects comparable. They have roughly the same size, are both originally developed in German, are today both multilingual, both have English translations, and, most important, despite being from two different domains, they have huge overlapping areas. Not least, both are freely available in RDF using SKOS.[11]

STW The STW Thesaurus for Economics provides vocabulary on any economic subject: more than 6,000 standardized subject headings (skos:Concepts, with preferred labels in English and German) and 19,000 additional keywords (skos:altLabels) in both languages. The vocabulary was developed for indexing purposes in libraries and economic research institutions and includes technical terms used in law, sociology, or politics, and geographic names. The entries are richly interconnected by 16,000 skos:broader/narrower and 10,000 skos:related relations. An additional hierarchy of main categories provides a high level overview. The vocabulary is maintained on a regular basis by ZBW[12], the German National Library of Economics - Leibniz Centre for Economics, and has been translated into SKOS [24].

TheSoz The Thesaurus for the Social Sciences (TheSoz) serves as a crucial instrument for indexing documents and research information in the social sciences. It contains overall about 12,000 keywords, from which 8,000 are standardized subject headings (in English and German) and 4,000 additional keywords. The thesaurus covers all topics and sub-disciplines of the social sciences. Additionally terms from associated and related disciplines are included in order to support an accurate and adequate indexing process of interdisciplinary, practical-oriented and multi-cultural documents. The thesaurus is owned and maintained by GESIS[13], the Leibniz Institute for the Social Sciences, and is available in SKOS [31].

Reference Alignment An alignment between STW and TheSoz already exists and has been manually created by domain experts in the KoMoHe project [18]. However, it does not cover the changes and enhancements in both thesauri since 2006. It is available in SKOS with the different matching types SKOS:exactMatch, SKOS:broaderMatch and SKOS:narrowerMatch. Within the reference alignment, concepts of one thesaurus are aligned to more than one concept of the second thesaurus. Thus, we face a *n:m* mapping of the concepts. All in all, 4,285 TheSoz concepts and 2,320 STW concepts are aligned with 2,839 exact matches, 34 broader matches and 1,416 narrower matches. It is important to note that the reference alignment only contains alignments between the descriptors of both thesauri, i.e., the concepts that are actually used for document indexing. The upper part of the hierarchy consists of non-descriptor concepts (or categories) that are only used to organize the descriptors below them. We take this specialty into account as we only assess the generated alignments between descriptors and ignore alignments between non-descriptors. However, this might change in the future, as the

[11] http://www.w3.org/2004/02/skos/
[12] http://zbw.eu/index-e.html
[13] http://www.gesis.org/en/home

results of this track could be used to extend the reference alignment to the upper part of the hierarchy.

Transformation Most ontology matching systems taking part in the OAEI only work on OWL ontologies and are not (yet) ready to deal with the specialties of a thesaurus. To get first results and to lower the barrier of taking part in this challenge, we provide OWL versions of the thesauri, generated as follows:

`skos:concept`	→	`owl:class`
`skos:prefLabel`, `skos:altLabel`	→	`rdfs:label`
`skos:scopeNote`, `skos:notation`	→	`rdfs:comment`
`A skos:narrower B`	→	`B rdfs:subClassOf A`
`skos:broader`	→	`rdfs:subClassOf`
`skos:related`	→	`rdfs:seeAlso`

This transformation obviously is not lossless. First and foremost, within the ontology, it is not recognizable which label is the preferred one and which ones are alternative labels. Since matching systems mostly have to focus on the labels, this transformation might lead to suboptimal results. There are, however, more fundamental differences between ontologies and thesauri that we show in the next section.

SKOS vs. OWL Thesauri – and other, similar knowledge structures like classifications or taxonomies – are often called lightweight ontologies [29]. However, ontologies and thesauri fundamentally differ. This is also reflected by the fact that with SKOS a specific model for thesauri exists that is formulated in OWL. There, a `skos:Concept` is not an `owl:Class`. Concepts sometimes represent classes, for example the STW concept COMMODITIES. However, this is not true for every `skos:Concept`, e.g., the STW concept GERMANY is an instance, not a class.

Having a look at the subordinate concepts of COMMODITIES, they mostly indeed represent classes, like METALS – METAL PRODUCTS – RAZOR. Nevertheless, the relation in SKOS between these concepts is `skos:broader`, not `rdfs:subClassOf`. A subclass relationship states that if a class *B* is a subclass of a class *A*, then all instances of *B* will also be instances of *A*. Here, all metals are commodities, but not all metal products are metals: the razor consists partly of metal, but it is no metal.

Thesauri are created for a very specific purpose and are used in a predetermined way. This is inter alia reflected by the distinction of descriptors and non-descriptors. Only descriptors are assigned to publications during the indexation or classification. All non-descriptors serve as additional information to provide the correct context or to build up a proper hierarchy. Such a distinction typically does not exist in an ontology.

Very difficult for ontology matchers (not necessarily only automatic ones) is the quasi-synonymy of the describing labels for a concept. A `skos:altLabel` is often used to indicate subconcepts that should be subsumed under the concept in question to avoid extensive subclassing. As an example, the STW descriptor *14117-2* with the preferred English label "Tropical fruit" has German alternative labels like "pineapple",

"avocado", and "kiwi". In an (OWL) ontology, these alternative labels should be modeled as subclasses of the class TROPICAL FRUIT. In contrast, other alternative labels might really indicate alternative, synonymous terms for the preferred label.

At last, instead of arbitrary semantic relations that are part of an ontology, in thesauri, relations like `skos:related` or `compoundEquivalence` in TheSoz exist. They often contain information for the (manual) use of the thesaurus for indexing, i.e., which descriptor should be used in which case or how combinations of descriptors are to be used. Transferring them to ontological relations is not always possible and depends often on the single case.

It can be seen that the development of a thesaurus matcher is indeed a challenge that differs from ontology matching. Nevertheless, the commonalities between thesauri and ontologies are large enough to pave the way for further developments by means of current ontology matchers.

7.2 Experimental Setting

To compare the created alignments with the reference alignment, we use the Alignment API. For this first evaluation, we only included equivalence relations (`skos:exactMatch`).

All matching processes have been performed on a Debian machine with one 2.4GHz core and 7GB RAM allocated to each system. The evaluation has been executed by using SEALS technologies. For ServOMap, ServOMapLt and Optima, we used slightly adapted ontologies as input since they cannot handle URIs with the last part only consisting of numbers as it is the case in the official version. Each participating system uses the OWL version. We computed precision, recall and F-measure ($\beta = 1$) for each matcher. We only consider equivalence correspondences between two descriptors as non-descriptors are not included in the reference alignment. This filtering improves the precision ($\approx 8\%$) as well as the F-measure ($\approx 4\%$) for all systems. Moreover, we measured the runtime, the size of the created alignment and checked whether a 1:1 alignment has been created. To assess the results of the matchers, we developed three straightforward matching strategies, using the original SKOS version of the thesauri:

- *Matcher*$_{prefDE}$: Compares the German lower-case preferred labels and generates a correspondence if these labels are completely equivalent.
- *Matcher*$_{prefEN}$: Compares the English lower-case preferred labels and generates a correspondence if these labels are completely equivalent.
- *Matcher*$_{pref}$: Creates a correspondence, if either *Matcher*$_{prefDE}$ or *Matcher*$_{prefEN}$ or both create a correspondence.
- *Matcher*$_{allLabels}$: Creates a correspondences whenever at least one label (preferred or alternative, all languages) of an entity is equivalent to one label of another entity.

7.3 Results

All systems listed in Table 9 are sorted according to their F-measures. Altogether 13 of the 21 submitted matching systems were able to create an alignment. Three matching

System	Precision	F-measure	Recall	Time [s]	Size	1:1
Matcher$_{pref}$	0.820	0.720	0.642	75	2190	-
Matcher$_{prefDE}$	0.891	0.717	0.601	42	1885	-
Matcher$_{allLabels}$	0.544	0.677	0.896	735	4605	-
GOMMA	0.537	0.674	0.906	804	4712	-
ServOMapLt	0.654	0.670	0.687	45	2938	-
LogMap	0.688	0.665	0.644	95	2620	-
ServOMap	0.717	0.665	0.619	44	2413	√
YAM++	0.595	0.664	0.750	496	3522	-
LogMapLt	0.577	0.662	0.776	21	3756	-
Hertuda	0.465	0.619	0.925	14363	5559	-
WeSeE	0.612	0.609	0.607	144070	2774	√
HotMatch	0.645	0.608	0.575	14494	2494	√
Matcher$_{prefEN}$	0.808	0.569	0.439	36	1518	-
CODI	0.434	0.445	0.481	39869	3100	√
MapSSS	0.520	0.272	0.184	2171	989	√
AROMA	0.107	0.184	0.652	1096	17001	-
Optima	0.321	0.117	0.072	37457	624	-

Table 9. Results of the Library track.

systems (MaasMatch, MEDLEY, Wmatch) did not finish within the time frame of one week while five threw an exception (no heap space exception).

Of all these systems, GOMMA performs best in terms of F-measure, closely followed by ServOMapLt and LogMap. However, the precision and recall measures vary a lot across the three systems. Depending on the application, an alignment either achieving high precision or recall is preferable. If recall is in the focus, the alignment created by GOMMA is probably the best choice with a recall of about 90%. Other systems generate alignments with higher precision, e.g. ServOMap with over 70% precision, while mostly having significantly lower recall values (except for Hertuda).

From the results obtained by the matching strategies taking the different types of labels into account, we can see that a matching based on preferred labels only, outperforms other matching strategies. *Matcher*$_{pref}$ achieves the highest F-measure in these tests. The results of *Matcher*$_{prefDE}$ and *Matcher*$_{prefEN}$ provide an insight into the language characteristics of both thesauri and the reference alignments. *Matcher*$_{prefDE}$ achieves the highest precision value (nearly 90%), albeit with a recall of only 60%. Both thesauri as well as the reference alignment have been developed in Germany and focus on German terms. From the results of *Matcher*$_{prefEN}$, we can see the difference: precision and especially recall significantly decrease when only the preferred English labels are used. On the one hand, only about 80% of the found correspondences are correct and on the other hand, less than a half of all correspondences can be found this way. This can be a disadvantage for systems that use NLP techniques on English labels or rely on language-specific background knowledge like WordNet.

The high precision values of the $pref*$ matchers reflect the fact that the preferred labels are chosen specifically to unambiguously identify the concepts. Our interpretation is that the English translations are partly not as precise as the original German

terms (drop in precision) and not consistent regarding the English terminology (drop in recall).

In contrast, the *Matcher*$_{allLabels}$ achieves a quite high recall (90%) but a rather low precision (54%). This means that most but not all of the correspondences can be found by only having a look at equivalent labels. However, when following this idea, nearly a half of the found correspondences are incorrect. The rather high F-measure of *Matcher*$_{allLabels}$ is therefore misleading, as at least if the results would be used unchecked in an retrieval system, a higher precision would clearly be preferred over a higher recall. In this respect, matchers like ServOMap show better results. In any case, it can be seen that a matching system using the original SKOS version could achieve a better result. The information loss when converting SKOS to OWL really matters.

Concerning runtime, LogMap as well as ServOMap are quite fast with a runtime below 50 seconds. These values are comparable or even better (LogMapLt) than both strategies computing the equivalence between preferred labels. Thus, they are very effective in matching large ontologies while achieving very good results. Other matchers take several hours or even days and do not produce better alignments in terms of F-measure. By computing the correlation between F-measure and runtime, we notice a slightly negative correlation (-0.085) but the small amount of samples is not sufficient to make a significant statement. However, we can say for certain that a longer runtime does not necessarily lead to better results.

We further observe that the *n:m* reference alignment affects the results because some matching systems (ServOMap, WeSeE, HotMatch, CODI, MapSSS) only create 1:1 alignments and discard correspondences with entities that already occur in another correspondence. Whenever a system creates a lot of n:m correspondences, e.g., Hertuda and GOMMA, the recall significantly increases. This difference becomes clear when comparing ServOMapLt and ServOMap. Both systems are mostly based on the same methods but ServOMapLt does not use the 1:1 filtering. Consequently, its recall increases and its precision decreases.

Since the reference alignment has not been updated for about six years, it does not contain updates of both thesauri. Thus, new correct correspondences might be found by matching systems but they are indicated as incorrect because they are not included in the reference alignment. Therefore, we applied a manual evaluation to check whether matching systems found correct correspondences which are not included in the reference alignment at all. In turn, these information can help to improve the reference alignment.

The manual evaluation has been conducted by domain experts. Many newly detected correspondences, which have not been contained in the reference alignment yet, have been considered. By now, we only examined correspondences between descriptors as well as the ones that did not contain a term which is already matched in the reference alignment.

The matchers detected between 38 and 251 correct correspondences, which have not been in the reference alignment before. This includes especially terms, which hold a strong syntactical similarity or equivalence. But, some matching systems even detected difficult correspondences, e.g., between the German label for “automated production” (“Automatische Produktion”) and “CAM”, which has been identified by their associated

non-preferred labels. Furthermore, correspondences of geographical terms have been detected, but some of the matchers have not been able to distinguish between the terms for citizens of a country, their language or the country itself, although these differences can be derived from the structure of the thesauri.

But this manual evaluation exposed several issues, which can either be explained by the typical behavior of matching systems or by domain-specific differences inside the thesauri. There are similar terms inside TheSoz and STW, which are used in totally different contexts, e.g., the term "self-assessment". Even when considering the structure of both thesauri these differences are difficult to identify. In general, term similarities often led to wrong correspondences, which is not surprising at first. But, in turn syntactically equal terms have not been detected simultaneously in some cases. By now, we did not have the possibility to evaluate the matching systems with the improved reference alignment, but we plan to perform this additional evaluation soon.

7.4 Conclusion

Nevertheless, the newly detected correspondences determine already a useful result for the maintainers of the two thesauri. The correct correspondences can be added to the existing reference alignment, which is already applied in information portals for supporting search term recommendation and query expansion services among differently indexed databases. As all matching systems delivered exact matches for the correspondences, some of the wrong correspondences will be examined again in the future, whether other relationships like broader, narrower or related matches can be considered for those.

We expect further improvements, if the matchers are tailored more specifically to the library track, i.e., if they exploit the information found in the original SKOS version. A promising approach is also the use of additional knowledge, e.g., instance data – resources that are indexed with different thesauri [30].

This time, we collected the results of the matchers as a first survey and compared them to our simple string-matching strategy that takes advantage of the different types of labels. In future evaluations, we assume that better results can be achieved and that these strategies simply form a baseline.

8 Large biomedical ontologies

This track aims at finding alignments between the large and semantically rich biomedical ontologies FMA, SNOMED CT, and NCI, which contains 78,989, 306,591 and 66,724 classes, respectively.

8.1 Test data

UMLS Metathesaurus [2] has been selected as the basis for the track's reference alignments. UMLS is currently the most comprehensive effort for integrating independently-developed medical thesauri and ontologies, including FMA, SNOMED CT, and NCI. Although the standard UMLS distribution does not directly provide "alignments" (in the

OAEI sense) between the integrated ontologies, it is relatively straightforward to extract them from the information provided in the distribution files (see [17] for details).

It has been noticed, however, that although the creation of UMLS alignments combines expert assessment and auditing protocols they lead to a significant number of logical inconsistencies when integrated with the corresponding source ontologies [17].

To address this problem, we have considered two refinements of the UMLS alignments that do not lead to (many) unsatisfiable classes. These refinements have been generated using LogMap's repair facility [16] and the Alcomo debugging system [20].

The track has been split into three matching problems: FMA-NCI, FMA-SNOMED and SNOMED-NCI; and each matching problem in three tasks involving different fragments of the input ontologies.

FMA-NCI matching We have compared the results of the matching tools against both the original and refined UMLS alignment sets:

- Original UMLS alignments: 3,024 alignments ($\equiv$).
- Refined UMLS alignments:
  - LogMap's repair module : 2,898 alignments ($\equiv$, $\sqsubseteq$, $\sqsupseteq$).
  - Alcomo debbuging system: 2,819 alignments ($\equiv$).

Three tasks have been considered involving different fragments of FMA and NCI:

- *Task 1* consists of matching two (relatively small) modules of FMA (3,696 classes, 5%) and NCI (6,488 classes 10%).
- *Task 2* consists of matching two (relatively large) modules of FMA (28,861 classes, 37%) and NCI (25,591 classes, 38%).
- *Task 3* consists of matching the whole FMA and NCI ontologies.

FMA-SNOMED matching We have compared the results of the matching tools against both the original and refined UMLS alignment sets:

- Original UMLS alignments: 9,008 alignments ($\equiv$).
- Refined UMLS alignments:
  - LogMap's repair module : 8,111 alignments ($\equiv$, $\sqsubseteq$, $\sqsupseteq$).
  - Alcomo debbuging system: 8,132 alignments ($\equiv$).

Three tasks have been considered involving different fragments of FMA and SNOMED:

- *Task 4* consists of matching two (relatively small) modules of FMA (10,157 classes, 13%) and SNOMED (13,412 classes, 5%).
- *Task 5* consists of matching two (relatively large) modules of FMA (50,523 classes, 64%) and SNOMED (122,464 classes, 40%).
- *Task 6* consists of matching the whole FMA and SNOMED ontologies.

SNOMED-NCI matching We have compared the results of the matching tools against both the original and refined UMLS alignment sets:

- Original UMLS alignments: 18,844 alignments ($\equiv$).
- Refined UMLS alignments:
  - LogMap's repair module : 18,324 alignments ($\equiv$, $\sqsubseteq$, $\sqsupseteq$).

Note that, at the time of creating the datasets, we could not compute a refined UMLS alignment set with Alcomo. The new version of Alcomo, however, has shown to be able to cope with SNOMED-NCI. Three tasks have been considered involving different fragments of SNOMED and NCI:

- *Task 7* consists of matching two (relatively small) modules of SNOMED (51,128 classes, 17%) and NCI (23,958 classes, 36%).
- *Task 8* consists of matching two (relatively large) modules of SNOMED (122,464 classes, 40%) and NCI (49,795 classes, 75%).
- *Task 9* consists of matching the whole SNOMED and NCI ontologies.

8.2 Results

We have run the evaluation in a high performance server with 16 CPUs and allocating 15GB RAM. In total, 15 out of 23 participating systems/configurations have been able to cope with at least one of the tasks of the track matching problem. Optima and MEDLEY failed to complete the smallest task with a time out of 24 hours, while OMR, OntoK, ASE and WeSeE, threw an Exception during the matching process. CODI was evaluated in a different setting using only 7GB and threw an exception related to insufficient memory when processing the smallest matching task. TOAST was not evaluated since it was only configured for the Anatomy track and it required a complex installation. LogMapLt, a very fast string matcher, has been used as baseline.

Note that GOMMA has also been evaluated with a configuration that exploits specialized background knowledge based on the UMLS Metathesaurus (GOMMA_{bk}). GOMMA_{bk} exploits the alignments $\mathcal{O}_1$-UMLS and UMLS-$\mathcal{O}_2$ and applies alignment composition techniques. LogMap, MaasMatch and YAM++ also use different kinds of background knowledge. LogMap uses normalisations and spelling variants from the domain specific resource UMLS Lexicon[14]. YAM++ and MaasMatch use the general purpose background knowledge provided by WordNet[15].

LogMap has also been evaluated with two configurations. LogMap's default algorithm computes an estimation of the overlapping between the input ontologies before the matching process, while the variant $\text{LogMap}_{\text{noe}}$ has this feature deactivated.

Precision and recall in Tables 11-13 average the obtained results with respect to the reference alignments. Systems have been ordered in terms of the average F-measure.

[14] http://www.nlm.nih.gov/pubs/factsheets/umlslex.html
[15] http://wordnet.princeton.edu/

Alignment coherence We have evaluated the coherence of the generated alignments and we have reported (1) number of unsatisfiabilities when reasoning[16] with the input ontologies together with the computed alignments, (2) the degree of unsatisfiable classes with respect to the size of the merged ontology (based on the Unsatisfiability Measure proposed in [22]), and (3) an approximation of the root unsatisfiability. The root unsatisfiability aims at providing a more precise amount of errors, since many of the unsatisfiabilities may be derived (i.e., a subclass of an unsatisfiable class will also be reported as unsatisfiable). The provided approximation is based on LogMap's (incomplete) repair facility and shows the number of classes that this facility needed to repair in order to solve (most of) the unsatisfiabilities [16].

LogMap and its variant $\text{LogMap}_{\text{noe}}$ were the unique systems generating an almost clean output. Tables 11-13 shown that even the most precise alignment sets may lead to a huge amount of unsatisfiable classes. This proves the importance of using techniques to assess the coherence of the generated alignments.

Note that, LogMap may fail to detect and repair unsatisfiable classes which are outside the computed overlapping (i.e. ontology fragments) between the input ontologies. Thus, $\text{LogMap}_{\text{noe}}$ provides, in general, a cleaner output than LogMap.

Runtimes Table 10 shows which systems were able to complete each of the matching tasks in less than 24 hours and the required computation times. Systems have been ordered with respect to the number of completed tasks and total time required to complete them. The last column reports the number of tasks that a system could complete. For example, only eight systems were able to complete all nine tasks. The last row shows the number of systems that could finish each of the tasks. The tasks involving larger ontology sizes were completed by only 8-10 systems. Furthermore, the tasks involving SNOMED were also harder with respect to both computation times and the number of systems that completed the tasks.

The runtimes were, in general, positive. For example, 7 systems completed Task 3 in less than 5 minutes. Additionally, the computation times for these systems increased "smoothly" with respect to the size of the input ontologies.

FMA-NCI matching Table 11 summarizes the results for the three tasks in the FMA-NCI matching problem. GOMMA_{bk} provided the best results in terms of F-measure in Task 1, whereas YAM++ in Tasks 2 and 3. GOMMA_{bk} also obtained the best results in terms of recall in all three tasks, while ServOMap computed the most precise alignments. Overall, the results were very positive and 7 systems obtained an F-measure greater than 0.80 in all three tasks.

LogMap and $\text{LogMap}_{\text{noe}}$ provided the same results in Task 1 since the input ontologies are already small fragments of FMA and NCI and thus, the overlapping estimation performed by LogMap did not have any impact.

[16] We have used HermiT [23] in the FMA-NCI and FMA-SNOMED matching problems. In the SNOMED-NCI matching problem we have estimated the number of unsatisfiable classes with the Dowling-Gallier algorithm for propositional Horn satisfiability [5] (implemented in LogMap's repair facility) since no OWL 2 reasoner is known to cope with the integration of SNOMED and NCI via alignments [15].

System	FMA-NCI			FMA-SNOMED			SNOMED-NCI			#
	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6	Task 7	Task 8	Task 9	
LogMapLt	8	29	55	14	96	171	54	104	178	9
ServOMapL	20	95	251	39	234	517	147	363	738	9
ServOMap	25	98	204	46	315	532	153	282	654	9
LogMap	18	77	131	65	484	612	221	514	955	9
$\text{LogMap}_{\text{noe}}$	18	74	206	63	521	791	211	575	1,505	9
GOMMA	26	69	217	54	437	1,994	197	527	1,820	9
GOMMA_{bk}	26	83	231	148	636	1,893	226	638	1,940	9
YAM++	78	245	1,304	326	3,780	23,900	1,901	6,127	30,155	9
AROMA	63	7,538	-	51,191	62,801	-	15,624	-	-	5
MapSSS	561	30,575	-	3,129	-	-	27,381	-	-	4
Hertuda	3,327	-	-	17,625	-	-	-	-	-	2
HotMatch	4,271	-	-	31,718	-	-	-	-	-	2
MaasMatch	27,157	-	-	-	-	-	-	-	-	1
AUTOMSv2	62,407	-	-	-	-	-	-	-	-	1
Wmatch	65,399	-	-	-	-	-	-	-	-	1
Completed	**15**	**10**	**8**	**12**	**9**	**8**	**10**	**8**	**8**	**88**

Table 10. System runtimes (s) and task completion.

In Task 1, our baseline also provided very good results in terms of F-measure and outperformed 8 of the participating systems. MaasMatch and Hertuda provided competitive results in terms of recall, but the low precision damaged the final F-measure. MapSSS and AUTOMSv2 provided a set of alignments with high precision, however, the F-measure was damaged due to the low recall of their alignments.

Efficiency in Task 2 and Task 3 have decreased considerably with respect to Task 1. This is mostly due to the fact that larger ontologies also involves more possible candidate alignments and it is harder to keep high precision values without damaging recall, and vice versa.

FMA-SNOMED matching Table 12 summarizes the results for the three tasks in the FMA-SNOMED matching problem. GOMMA_{bk} provided the best results in terms of F-measure in Task 4, whereas ServOMap in Tasks 5 and 6. GOMMA_{bk} also obtained the best results in terms of recall in all three tasks, while LogMapLt computed the most precise alignments in Task 4 and ServOMapL in Tasks 5 and 6.

Overall, the results were less positive than in the FMA-NCI matching tasks and only 5 systems obtained an F-measure greater than 0.70 in all three tasks. Furthermore, 6 systems (including our baseline) failed to provide a recall higher than 0.4. Thus, matching FMA against SNOMED represents a significant leap in complexity with respect to the FMA-NCI matching problem.

As in the FMA-NCI matching problem, efficiency also decreases as the ontology size increases. The most important variations were suffered by GOMMA_{bk} and GOMMA, where their average precision decreased from 0.893 and 0.875 (Task 4) to 0.571 and 0.389 (Task 5), respectively. This is an interesting fact, since the background knowledge used by GOMMA_{bk} could not avoid the decrease in precision while keeping the highest recall.

SNOMED-NCI matching Table 13 summarizes the results for the three matching tasks in the SNOMED-NCI matching problem. $\text{LogMap}_{\text{noe}}$ provided the best results in terms of both recall and F-measure in Tasks 7 and 8 while YAM++ obtained the

System	Time (s)	Size	Average			Incoherence		
			P	F	R	All Unsat.	Degree	Root Unsat.
Task 1: small FMA and NCI fragments								
$GOMMA_{bk}$	26	2,843	0.94	0.92	0.91	6,204	61%	193
YAM++	78	2,614	0.96	0.91	0.86	2,352	23%	92
LogMap/$LogMap_{noe}$	18	2,740	0.93	0.90	0.88	2	0,02%	0
GOMMA	26	2,626	0.95	0.90	0.86	2,130	21%	127
ServOMapL	20	2,468	0.96	0.88	0.82	5,778	57%	79
LogMapLt	8	2,483	0.95	0.87	0.81	2,104	21%	116
ServOMap	25	2,300	0.97	0.86	0.77	5,597	55%	50
HotMatch	4,271	2,280	0.96	0.84	0.75	285	3%	65
Wmatch	65,399	3,178	0.79	0.82	0.86	3,168	31%	482
AROMA	63	2,571	0.86	0.80	0.76	7,196	70%	421
Hertuda	3,327	4,309	0.58	0.69	0.86	2,675	26%	277
MaasMatch	27,157	3,696	0.61	0.68	0.78	9,598	94%	3,113
AUTOMSv2	62,407	1,809	0.80	0.62	0.50	5,346	52%	392
MapSSS	561	1,483	0.84	0.57	0.43	565	6%	94
Task 2: big FMA and NCI fragments								
YAM++	245	2,688	0.90	0.87	0.83	22,402	35%	102
ServOMapL	95	2,640	0.89	0.85	0.81	22,315	35%	143
GOMMA	69	2,810	0.86	0.84	0.83	2,398	4%	116
$GOMMA_{bk}$	83	3,116	0.81	0.84	0.87	4,609	8%	146
$LogMap_{noe}$	74	2,663	0.87	0.83	0.80	5	0.01%	0
LogMap	77	2,656	0.87	0.83	0.79	5	0.01%	0
ServOMap	98	2,413	0.91	0.83	0.76	21,688	34%	86
LogMapLt	29	3,219	0.73	0.77	0.81	12,682	23%	443
AROMA	7,538	3,856	0.54	0.61	0.69	20,054	24%	1600
MapSSS	30,575	2,584	0.38	0.36	0.34	21,893	40%	358
Task 3: whole FMA and NCI ontologies								
YAM++	1,304	2,738	0.89	0.86	0.83	50,550	29%	141
GOMMA	217	2,843	0.85	0.84	0.83	5,574	4%	139
ServOMapL	251	2,700	0.87	0.84	0.81	50,334	28%	164
$GOMMA_{bk}$	231	3,165	0.80	0.83	0.87	12,939	9%	245
$LogMap_{noe}$	206	2,646	0.87	0.83	0.79	9	0.01%	0
LogMap	131	2,652	0.86	0.82	0.78	9	0.01%	0
ServOMap	204	2,465	0.89	0.82	0.76	48,743	27%	114
LogMapLt	55	3,466	0.68	0.74	0.81	26,429	9%	778

Table 11. Results for the FMA-NCI matching problem.

best results in Task 9. $GOMMA_{bk}$ obtained the best recall in Task 9 and ServOMap generated the most precise alignments in all three tasks.

As in the previous matching problems, efficiency decreases as the ontology size increases. For example, in Task 9 none of the systems could reach an F-measure of 0.7, while seven systems (including our baseline) exceeded this value in Task 7.

8.3 Conclusions

Although the proposed matching tasks represented a significant leap in complexity with respect to the tasks in previous campaigns, the obtained results have been very promising and eight systems completed all matching tasks with very competitive results.

There is, however, plenty of room for improvement: (1) most of the participating systems disregard the coherence of the generated alignments; (2) the size of the input ontologies should not significantly affect efficiency; and (3) recall in the tasks involving SNOMED should be improved while keeping the current precision values.

The alignment coherence measure was the weakest point of the systems participating in this track. As shown in Tables 11-13, even highly precise alignment sets may

System	Time (s)	Size	Average			Incoherence		
			P	F	R	All Unsat.	Degree	Root Unsat.
Task 4: small FMA and SNOMED fragments								
$GOMMA_{bk}$	148	8,598	0.89	0.90	0.91	13,685	58%	4,674
ServOMapL	39	6,346	0.92	0.79	0.69	10,584	45%	3,056
YAM++	326	6,421	0.91	0.79	0.69	14,534	62%	3,150
$LogMap_{noe}$	63	6,363	0.91	0.78	0.69	0	0%	0
LogMap	65	6,164	0.91	0.77	0.67	2	0.01%	2
ServOMap	46	6,008	0.92	0.76	0.66	8,165	35%	2,721
GOMMA	54	3,667	0.88	0.53	0.38	2,058	9%	206
MapSSS	3,129	3,458	0.75	0.44	0.31	9,084	39%	389
AROMA	51,191	5,227	0.53	0.40	0.33	21,083	89%	2,296
HotMatch	31,718	2,139	0.84	0.34	0.21	907	4%	104
LogMapLt	14	1,645	0.94	0.31	0.18	773	3%	21
Hertuda	17,625	3,051	0.56	0.30	0.20	1,020	4%	47
Task 5: big FMA and SNOMED fragments								
ServOMapL	234	6,563	0.88	0.77	0.69	55,970	32%	1,192
ServOMap	315	6,272	0.88	0.75	0.65	143,316	83%	1,320
YAM++	3,780	7,003	0.82	0.75	0.68	69,345	40%	1,360
$LogMap_{noe}$	521	6,450	0.84	0.73	0.64	0	0%	0
LogMap	484	6,292	0.83	0.71	0.62	0	0%	0
$GOMMA_{bk}$	636	12,614	0.57	0.68	0.86	75,910	44%	3,344
GOMMA	437	5,591	0.39	0.31	0.26	7,343	4%	480
AROMA	62,801	2,497	0.66	0.30	0.20	54,459	31%	271
LogMapLt	96	1,819	0.85	0.30	0.18	2,994	2%	24
Task 6: whole FMA and SNOMED ontologies								
ServOMapL	517	6,605	0.88	0.77	0.69	99,726	26%	2,862
ServOMap	532	6,320	0.87	0.75	0.65	273,242	71%	2,617
YAM++	23,900	7,044	0.81	0.74	0.68	106,107	28%	3,393
LogMap	612	6,312	0.83	0.71	0.62	10	0.003%	0
$LogMap_{noe}$	791	6,406	0.82	0.71	0.62	10	0.003%	0
$GOMMA_{bk}$	1,893	12,829	0.56	0.68	0.86	119,657	31%	5,289
LogMapLt	171	1,823	0.85	0.30	0.18	4,938	1%	37
GOMMA	1,994	5,823	0.35	0.29	0.24	10,752	3%	609

Table 12. Results for the FMA-SNOMED matching problem.

lead to a huge number of unsatisfiable classes. Thus, the use of techniques to assess mapping coherence is critical. Unfortunately, only a few systems in OAEI 2012 have successfully used such techniques. In future campaigns, this aspect should not be neglected. Developers can reuse available state-of-the-art mapping debugging techniques such as the implemented in Alcomo [20] or in LogMap [16].

The UMLS-based reference alignments may contain errors and may also be incomplete. Thus, in order to turn the current reference alignments into a agreed-upon gold standard, expert assessment is required, which is almost unfeasible for large alignment sets. In this track, we have opted to move towards a "silver standard" by "harmonising" the outputs of different matching tools over the different matching tasks (see `http://www.cs.ox.ac.uk/isg/projects/SEALS/oaei/harmonisation/oaei2012_harmo.html` for details).

System	Time (s)	Size	Average			Incoherence		
			P	F	R	All Unsat.	Degree	Root Unsat.
Task 7: small SNOMED and NCI fragments								
$LogMap_{noe}$	211	13,525	0.90	0.75	0.65	0	0%	0
LogMap	221	13,454	0.90	0.75	0.65	0	0%	0
$GOMMA_{bk}$	226	12,294	0.94	0.75	0.62	48,681	65%	863
YAM++	1,901	11,961	0.95	0.74	0.61	50,089	67%	471
ServOMapL	147	11,730	0.95	0.74	0.60	62,367	83%	657
ServOMap	153	10,829	0.97	0.71	0.56	51,020	68%	467
LogMapLt	54	10,947	0.95	0.70	0.56	61,269	82%	801
GOMMA	197	10,555	0.94	0.68	0.53	42,813	57%	851
AROMA	15,624	11,783	0.85	0.66	0.54	70,491	94%	1,286
MapSSS	27,381	9,608	0.79	0.54	0.41	46,083	61%	794
Task 8: big SNOMED and NCI fragments								
$LogMap_{noe}$	575	13,184	0.88	0.73	0.62	0	0%	0
YAM++	6,127	13,083	0.86	0.71	0.61	104,492	61%	618
ServOMapL	363	12,784	0.86	0.70	0.59	136,909	79%	1,101
LogMap	514	12,142	0.87	0.69	0.57	3	0.002%	2
ServOMap	282	11,632	0.89	0.69	0.56	110,253	64%	820
$GOMMA_{bk}$	638	15,644	0.72	0.66	0.61	116,451	68%	2,741
LogMapLt	104	12,741	0.81	0.66	0.56	131,073	76%	2,201
GOMMA	527	12,320	0.80	0.63	0.53	96,945	56%	1,621
Task 9: whole SNOMED and NCI ontologies								
YAM++	30,155	14,103	0.79	0.68	0.60	238,593	64%	979
ServOMapL	738	13,964	0.79	0.68	0.59	286,790	77%	1,557
LogMap	955	13,011	0.81	0.67	0.57	16	0.004%	10
$LogMap_{noe}$	1,505	13,058	0.81	0.67	0.57	0	0%	0
ServOMap	654	12,462	0.83	0.67	0.56	230,055	62%	1,546
$GOMMA_{bk}$	1,940	17,045	0.66	0.64	0.61	239,708	64%	4,297
LogMapLt	178	14,043	0.74	0.63	0.56	305,648	82%	3,160
GOMMA	1,820	13,693	0.71	0.61	0.53	215,959	58%	2,614

Table 13. Results for the SNOMED-NCI matching problem.

9 Instance matching

The instance matching track aims at evaluating the performance of different matching tools on the task of matching RDF individuals which originate from different sources but describe the same real-world entity. Data interlinking is known under many names according to various research communities: equivalence mining, record linkage, object consolidation and coreference resolution to mention the most used ones. In each case, these terms are used for the task of finding equivalent entities in or across data sets [14]. As the quantity of data sets published on the Web of data dramatically increases, the need for tools helping to interlink resources becomes more critical. It is particularly important to maximize the automation of the interlinking process in order to be able to follow this expansion.

Unlike the other tracks, the instance matching track specifically focuses on an ontology ABox. However, the problems which have to be resolved in order to correctly match instances can originate at the schema level (use of different properties and classification schemas) as well as at the data level, e.g., different formats of values. This year, the track included two tasks. The first task, called *Sandbox*, is a simple dataset that has been specifically conceived to provide a examples of some specific matching problems highlighted (like name spelling and other controlled variations). This is intended to serve as a test for those tools that are in an initial phase of their development

process and/or for tools that are facing very focused tasks, like for example person name matching. The second one, called *IIMB* is an OWL-based dataset that is automatically generated by introducing a set of controlled transformations in an initial OWL Abox.

The list of participants to the instance matching track is shown in Table 14.

Dataset	LogMap	LogMap_lite	SBUEI	semsim
Sandbox	√	√	√	
IIMB	√	√	√	√

Table 14. Participants in the instance matching track.

9.1 Sandbox

The dataset used for the Sandbox task has been automatically generated by extracting data from Freebase, an open knowledge base that contains information about 11 million real objects including movies, books, TV shows, celebrities, locations, companies and more. Data has been extracted in JSON through the Freebase JAVA API[17]. Sandox is a collection of OWL files consisting of 31 concepts, 36 object properties, 13 data properties and 375 individuals divided into 10 test cases[18]. In order to provide simple matching challenges mainly conceived for systems in their initial developing phase, we limited the way data are transformed from the original Abox to the test cases. In particular, we introduced only changes in data format (misspelling, errors in text, etc.).

Sandbox results An overview of the precision, recall and F_1-measure results of the Sandbox task is shown in Table 15.

test	Precision	Recall	F_1-measure
LogMap	0.94	0.94	0.94
LogMap_lite	0.95	0.89	0.92
SBUEI	0.95	0.98	0.96

Table 15. Results of the Sandbox task.

As expected, all the participating systems obtained very good results for the simple tests provided by the Sandbox task. This result confirms that the currently available systems for instance matching provide efficient facilities for data matching when dealing with simple errors and syntactic heterogeneities.

9.2 IIMB

The IIMB task is focused on two main goals:

1. to provide an evaluation data set for various kinds of data transformations, including value transformations, structural transformations and logical transformations;

[17] `http://code.google.com/p/freebase-java/`

[18] DL expressivity of ontologies is $\mathcal{ALHI}(D)$

2. to cover a wide spectrum of possible techniques and tools.

ISLab Instance Matching Benchmark (IIMB), that has been generated using the SWING tool [13]. Participants were requested to find the correct correspondences among individuals of the first knowledge base and individuals of the other one. An important task here is that some of the transformations require automatic reasoning for finding the expected alignments.

IIMB is composed of a set of test cases, each one represented by a set of instances, i.e., an OWL ABox, built from an initial data set of real linked data extracted from the web. Then, the ABox is automatically modified in several ways by generating a set of new ABoxes, called *test cases*. Each test case is produced by transforming the individual descriptions in the reference ABox in new individual descriptions that are inserted in the test case at hand. The goal of transforming the original individuals is twofold: on one side, we provide a simulated situation where data referring to the same objects are provided in different data sources; on the other side, we generate different data sets with a variable level of data quality and complexity. IIMB provides transformation techniques supporting modifications of data property values, modifications of number and type of properties used for the individual description, and modifications of the individuals classification. The first kind of transformations is called *data value transformation* and it aims at simulating the fact that data expressing the same real object in different data sources may be different because of data errors or because of the usage of different conventional patterns for data representation. The second kind of transformations is called *data structure transformation* and it aims at simulating the fact that the same real object may be described using different properties/attributes in different data sources. Finally, the third kind of transformations, called *data semantic transformation*, simulates the fact that the same real object may be classified in different ways in different data sources.

The 2012 edition has been created by exploiting the same OWL source used for the Sandbox task. The main difference is that we introduced in IIMB a large set of data transformations. In particular, test cases from 0 to 20 contain changes in data format (misspelling, errors in text, etc); test cases 21 to 40 contain changes in structure (properties missing, RDF triples changed); 41 to 60 contain logical changes (class membership changed, logical errors); finally, test cases 61 to 80 contain a mix of the previous.

IIMB results An overview of the precision, recall and F_1-measure results per set of tests of the IIMB subtrack is shown in Table 16. A precision-recall graph visualization is shown in Figure 5.

As a general comment, we can conclude that all the four systems participating in this edition of the instance matching track obtained good results, both in terms of precision and recall. Table 16 suggests that the most challenging tasks in IIMB this year were those included in test cases 061-080, which provides a combination of different data transformations, ranging from syntactic to semantic transformations. According to this conclusion, we will better investigate the problem of dealing with the problem of combining data transformations in order to generate a new challenging dataset for instance matching evaluation.

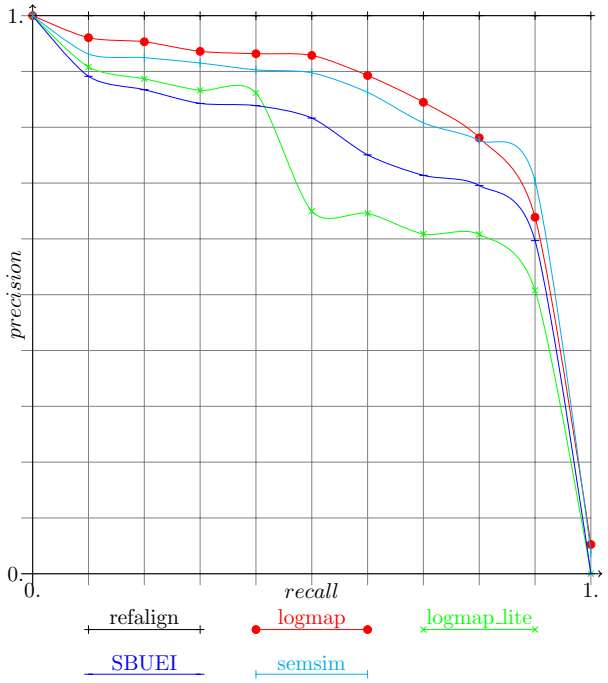


Fig. 5. Precision/recall of the systems participating in the IIMB subtrack.

test	001–020			021–040			041–060			061–080		
	P	F	R	P	F	R	P	F	R	P	F	R
LogMap	.94	.90	.87	.93	.96	1.0	.93	.96	1.0	.95	.86	.79
LogMap_lite	.95	.78	.66	.93	.95	.97	.74	.84	.98	.77	.73	.69
SBUEI	.96	.97	.98	.97	.98	.99	.91	.90	.89	.58	.53	.48
semsim	.93	.93	.93	.91	.91	.91	.94	.94	.94	.66	.66	.66

Table 16. Results of the IIMB subtrack.

10 Lesson learned and suggestions

There are, this year, very few comments about the evaluation execution:

A) This year indicated again that requiring participants to implement a minimal interface was not a strong obstacle to participation. Moreover, the community seems to get used to the SEALS technology introduced for OAEI 2011. This might be one of the reasons for an increasing participation.
B) We have not delivered any comparative results prior to the deadline for submitting the papers that contain the systems description. This procedure was motivated by avoiding a focus on competitive aspects of OAEI. However, several tool developers have complained about this procedure, because they wanted to include such results in their papers.
C) Last years we reported that we had many new participants. The same trend can be observed for 2012.
D) Again, given the high number of publications on data interlinking, it is surprising to have so few participants to the instance matching tracks.

11 Conclusions

These year, both in OAEI 2011.5 and 2012, some tracks have focused on scalability and runtime measurement. The low number of systems that could generate results for the Anatomy track was an uncommon result in 2011. However, in 2012 many more systems could generate results for the Anatomy track. Moreover, many systems could also generate results for the Library and the Large Biomed track that is concerned with significantly larger test cases.

Compared to the previous years, we observed a significant improvements of runtimes. Matching systems are becoming more robust and also more efficient with respect to runtimes. In particular, these improvements are more general compared to increased precision and recall scores. We dare thinking that this improvement has been steered by OAEI efforts towards more challenging test sets.

There is a high variance in runtimes and there is no correlation between runtime and quality of the generated results. This is a result that we already observed in 2011.

There has been a considerable increase in the number of participants implementing specific techniques for dealing with the task of matching ontologies in different natural languages (seven participants in 2012, three in OAEI 2011.5). Although there is room for improvements to achieve the same level of compliance than in the original OntoFarm dataset, this increase is a sign that the field is progressing.

All participants have provided a description of their systems and their experience in the evaluation. These OAEI papers, like the present one, have not been peer reviewed. However, they are full contributions to this evaluation exercise and reflect the hard work and clever insight people put in the development of participating systems. Reading the papers of the participants should help people involved in ontology matching to find what makes these algorithms work and what could be improved. Sometimes participants offer alternate evaluation results.

The Ontology Alignment Evaluation Initiative will continue these tests by improving both test cases and testing methodology for being more accurate. Further information can be found at:

`http://oaei.ontologymatching.org`.

Acknowledgements

We warmly thank the participants of this campaign. We know that they have worked hard for having their matching tools executable in time and they provided insightful papers presenting their experience. The best way to learn about the results remains to read the following papers.

We would like to thank Andreas Oskar Kempf from GESIS for the manual evaluation of the new detected correspondences. We thank Jan Noessner for providing data in the process of constructing the IIMB data set. We are grateful to Martin Ringwald and Terry Hayamizu for providing the reference alignment for the anatomy ontologies and thank Elena Beisswanger for her thorough support on improving the quality of the data set.

We also thank the other members of the Ontology Alignment Evaluation Initiative steering committee: Yannis Kalfoglou (Ricoh laboratories, UK), Miklos Nagy (The Open University (UK), Natasha Noy (Stanford University, USA), Yuzhong Qu (Southeast University, CN), York Sure (Leibniz Gemeinschaft, DE), Jie Tang (Tsinghua University, CN), George Vouros (University of the Aegean, GR).

José Luis Aguirre, Bernardo Cuenca Grau, Jérôme Euzenat, Ernesto Jimenez-Ruiz, Christian Meilicke, Heiner Stuckenschmidt and Cássia Trojahn dos Santos have been partially supported by the SEALS (IST-2009-238975) European project.

Ondřej Šváb-Zamazal has been supported by the CSF grant P202/10/0761.

Ernesto and Bernardo have been partially supported by the Royal Society, EPSRC project LogMap.

References

1. Benhamin Ashpole, Marc Ehrig, Jérôme Euzenat, and Heiner Stuckenschmidt, editors. *Proc. of the K-Cap Workshop on Integrating Ontologies*, Banff (Canada), 2005.
2. Olivier Bodenreider. The unified medical language system (UMLS): integrating biomedical terminology. *Nucleic Acids Research*, 32:267–270, 2004.
3. Caterina Caracciolo, Jérôme Euzenat, Laura Hollink, Ryutaro Ichise, Antoine Isaac, Véronique Malaisé, Christian Meilicke, Juan Pane, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Sváb-Zamazal, and Vojtech Svátek. Results of the ontology alignment evaluation initiative 2008. In *Proc. 3rd International Workshop on Ontology Matching (OM) collocated with ISWC*, pages 73–120, Karlsruhe (Germany), 2008.
4. Jérôme David, Jérôme Euzenat, François Scharffe, and Cássia Trojahn dos Santos. The alignment API 4.0. *Semantic web journal*, 2(1):3–10, 2011.
5. William F. Dowling and Jean H. Gallier. Linear-time algorithms for testing the satisfiability of propositional Horn formulae. *J. Log. Prog.*, 1(3):267–284, 1984.
6. Jérôme Euzenat, Alfio Ferrara, Laura Hollink, Antoine Isaac, Cliff Joslyn, Véronique Malaisé, Christian Meilicke, Andriy Nikolov, Juan Pane, Marta Sabou, François Scharffe,

Pavel Shvaiko, Vassilis Spiliopoulos, Heiner Stuckenschmidt, Ondrej Sváb-Zamazal, Vojtech Svátek, Cássia Trojahn dos Santos, George Vouros, and Shenghui Wang. Results of the ontology alignment evaluation initiative 2009. In *Proc. 4th Workshop on Ontology Matching (OM) collocated with ISWC*, pages 73–126, Chantilly (USA), 2009.

7. Jérôme Euzenat, Alfio Ferrara, Christian Meilicke, Andriy Nikolov, Juan Pane, François Scharffe, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Sváb-Zamazal, Vojtech Svátek, and Cássia Trojahn dos Santos. Results of the ontology alignment evaluation initiative 2010. In Pavel Shvaiko, Jérôme Euzenat, Fausto Giunchiglia, Heiner Stuckenschmidt, Ming Mao, and Isabel Cruz, editors, *Proc. 5th ISWC workshop on ontology matching (OM) collocated with ISWC, Shanghai (China)*, pages 85–117, 2010.
8. Jérôme Euzenat, Alfio Ferrara, Robert Willem van Hague, Laura Hollink, Christian Meilicke, Andriy Nikolov, François Scharffe, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Sváb-Zamazal, and Cássia Trojahn dos Santos. Results of the ontology alignment evaluation initiative 2011. In Pavel Shvaiko, Isabel Cruz, Jérôme Euzenat, Tom Heath, Ming Mao, and Christoph Quix, editors, *Proc. 6th ISWC workshop on ontology matching (OM), Bonn (DE)*, pages 85–110, 2011.
9. Jérôme Euzenat, Antoine Isaac, Christian Meilicke, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Svab, Vojtech Svatek, Willem Robert van Hage, and Mikalai Yatskevich. Results of the ontology alignment evaluation initiative 2007. In *Proc. 2nd International Workshop on Ontology Matching (OM) collocated with ISWC*, pages 96–132, Busan (Korea), 2007.
10. Jérôme Euzenat, Christian Meilicke, Pavel Shvaiko, Heiner Stuckenschmidt, and Cássia Trojahn dos Santos. Ontology alignment evaluation initiative: six years of experience. *Journal on Data Semantics*, XV:158–192, 2011.
11. Jérôme Euzenat, Malgorzata Mochol, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Svab, Vojtech Svatek, Willem Robert van Hage, and Mikalai Yatskevich. Results of the ontology alignment evaluation initiative 2006. In *Proc. 1st International Workshop on Ontology Matching (OM) collocated with ISWC*, pages 73–95, Athens, Georgia (USA), 2006.
12. Jérôme Euzenat and Pavel Shvaiko. *Ontology Matching*. Springer, Heidelberg (DE), 2007.
13. Alfio Ferrara, Stefano Montanelli, Jan Noessner, and Heiner Stuckenschmidt. Benchmarking matching applications on the semantic web. In *Proc. of the 8th Extended Semantic Web Conference (ESWC 2011)*, Heraklion, Greece, 2011.
14. Alfio Ferrara, Andriy Nikolov, and Francois Scharffe. Data linking for the semantic web. *International Journal on Semantic Web and Information Systems*, 7(3):46–76, 2011.
15. E. Jiménez-Ruiz, B. Cuenca Grau, and I. Horrocks. On the feasibility of using OWL 2 DL reasoners for ontology matching problems. In *OWL Reasoner Evaluation Workshop*, 2012.
16. Ernesto Jimenez-Ruiz and Bernardo Cuenca Grau. LogMap: Logic-based and scalable ontology matching. In *Int'l Sem. Web Conf. (ISWC)*, pages 273–288, 2011.
17. Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Ian Horrocks, and Rafael Berlanga. Logic-based assessment of the compatibility of UMLS ontology sources. *J. Biomed. Sem.*, 2, 2011.
18. Philipp Mayr and Vivien Petras. Building a terminology network for search: The KoMoHe project. In *Proc. of the Int. Conference on Dublin Core and Metadata Applications*, pages 177 – 182, 2008.
19. Christian Meilicke. *Alignment Incoherence in Ontology Matching*. PhD thesis, University Mannheim, 2011.
20. Christian Meilicke. *Alignment Incoherence in Ontology Matching*. PhD thesis, University of Mannheim, 2011.
21. Christian Meilicke, Raúl García-Castro, Fred Freitas, Willem Robert van Hage, Elena Montiel-Ponsoda, Ryan Ribeiro de Azevedo, Heiner Stuckenschmidt, Ondřej Šváb Zamazal, Vojtěch Svátek, Andrei Tamilin, Cássia Trojahn, and Shenghui Wang. MultiFarm: A benchmark for multilingual ontology matching. *Web Semantics: Science, Services and Agents on the World Wide Web*, (0):–, 2012.

22. Christian Meilicke and Heiner Stuckenschmidt. Incoherence as a basis for measuring the quality of ontology mappings. In *Proc. 3rd International Workshop on Ontology Matching (OM) collocated with ISWC*, pages 1–12, Karlsruhe (Germany), 2008.
23. Boris Motik, Rob Shearer, and Ian Horrocks. Hypertableau reasoning for description logics. *J. Artif. Intell. Res.*, 36:165–228, 2009.
24. Joachim Neubert. Bringing the "thesaurus for economics" on to the web of linked data. In *Proc. of the WWW Workshop on Linked Data on the Web (LDOW)*, 2009.
25. Maria Roşoiu, Cássia Trojahn dos Santos, and Jérôme Euzenat. Ontology matching benchmarks: generation and evaluation. In Pavel Shvaiko, Isabel Cruz, Jérôme Euzenat, Tom Heath, Ming Mao, and Christoph Quix, editors, *Proc. 6th International Workshop on Ontology Matching (OM) collocated with ISWC, Bonn (Germany)*, 2011.
26. Pavel Shvaiko and Jérôme Euzenat. Ontology matching: state of the art and future challenges. *IEEE Transactions on Knowledge and Data Engineering*, 2012, to appear.
27. York Sure, Oscar Corcho, Jérôme Euzenat, and Todd Hughes, editors. *Proc. of the Workshop on Evaluation of Ontology-based Tools (EON) collocated with ISWC*, Hiroshima (Japan), 2004.
28. Cássia Trojahn dos Santos, Christian Meilicke, Jérôme Euzenat, and Heiner Stuckenschmidt. Automating OAEI campaigns (first report). In *Proc. International Workshop on Evaluation of Semantic Technologies (iWEST) collocated with ISWC*, Shanghai (China), 2010.
29. Michael Uschold and Michael Gruninger. Ontologies and semantics for seamless connectivity. *SIGMOD Rec.*, 33(4):58–64, 2004.
30. Shenghui Wang, Antoine Isaac, Stefan Schlobach, Lourens van der Meij, and Balthasar Schopman. Instance-based semantic interoperability in the cultural heritage. *Semantic Web*, 3(1):45–64, 2012.
31. Benjamin Zapilko, Johann Schaible, Philipp Mayr, and Brigitte Mathiak. TheSoz: A SKOS representation of the thesaurus for the social sciences. *Semantic Web – Interoperability, Usability, Applicability*, 2012. accepted.

Grenoble, Milano, Amsterdam, Mannheim, Milton-Keynes, Montpellier, Trento,
Prague, Toulouse, Köln, Oxford,
December 2012

ASE Results for OAEI 2012

Konstantinos Kotis[1], Artem Katasonov[1], Jarkko Leino[1]

[1] VTT Technical Research Centre of Finland, Tampere, FI
{Ext-konstantinos.kotis, Artem.Katasonov,
Jarkko.Leino}@vtt.fi

Abstract. This paper presents ASE (Aligning Smart Entities) tool for the automated alignment of OWL domain ontology definitions in the context of Internet of Things (IoT). The effort is based on experience gained by the development of AUTOMSv2 for OAEI 2012. The development process of this tool has been driven by our motivation to use the ontology alignment functionality as part of the Smart Proxy approach for the matchmaking of IoT entities. More specifically, ASE supports the automated deployment of applications on environments that IoT devices (sensors and actuators) have been already deployed. This paper presents the alignment approach followed towards developing the tool and the official results obtained for OAEI 2012 campaign.

1 Presentation of the system

1.1 State, purpose, general statement

ASE (Aligning Smart Entities) is an automated ontology alignment tool based on AUTOMSv2 tool (http://ai-lab-webserver.aegean.gr/kotis/AUTOMSv2), a baseline tool we have developed for OAEI 2012 campaign. It computes 1:1 (one to one) alignments of two input domain ontologies in OWL, discovering equivalence and subsumption axioms between ontology elements, both classes and properties. The features that this tool integrates are summarized in the following points:

- It is implemented with the widely used open source Java Alignment API [1]
- It synthesizes lexical and lexicon-based alignment methods, using union aggregation operator
- It integrates state-of-the-art alignment methods with standard and extended methods from the Java Alignment API
- Implements a language translation method for non-English ontology elements

Comparing with AUTOMSv2, in ASE

This work was carried out during the tenure of an ERCIM "Alain Bensoussan" Fellowship Programme. This Programme is supported by the Marie Curie Co-funding of Regional, National and International Programmes (COFUND) of the European Commission

a) We do not implement a profiling and configuration strategy, but instead we use a fixed synthesis method based on experience and observation of AUTOMSv2 behavior and also on specific performance requirements that the application domain of IoT and the specific Smart Proxy approach have been implied,
b) We implement the discovery of subsumption relations between concept/property pairs, in addition to equivalences,
c) We implement a new method for translating Non-English ontologies, a method that is based on the Microsoft Bing Translator API
d) We implement some utility functions for handling compound terms

The problem of computing alignments between ontologies can be formally described as follows: Given two ontologies $O_1 = (S_1, A_1)$, $O_2 = (S_2, A_2)$ (where S_i denotes the signature and A_i the set of axioms that specify the intended meaning of terms in S_i) and an element (class or property) E_i^1 in the signature S_1 of O_1, locate a corresponding element E_j^2 in S_2, such that a mapping relation (E_i^1, E_j^2, r) holds between them. r can be any relation such as the equivalence ($\equiv$) or the subsumption ($\sqsubseteq$) axiom or any other semantic relation e.g. meronym. For any such correspondence a mapping method may relate a value γ that represents the preference to relating E_i^1 with E_j^2 via r. If there is not such a preference, we assume that the method equally prefers any such assessed relation for the element E_1. The correspondence is denoted by (E_i^1, E_j^2, r, γ). The set of computed mapping relations produces the mapping function $f:S_1 \rightarrow S_2$ that must preserve the semantics of representation: i.e. all models of axioms A_2 must be models of the translated A_1 axioms: i.e. $A_2 \models f(A_1)$.

ASE can be seen as a subversion of AUTOMSv2 ontology alignment tool, in the sense that it uses a specific synthesis configuration of AUTOMSv2 alignment methods. The synthesis of alignment methods that exploit different types of information may discover different types of relations between elements have been already proved to be of great benefit [2, 5]. ASE configuration is based on the requirement that the related input ontology definitions in the application domain that this tool is used are very often flat (no structure), have no instances (unpopulated), have very few concepts/properties (1 to 5 in most cases), have no expressive axioms and compound terms are very common.

In ASE we follow a modern synthesis strategy, which performs composition of results at different levels: the resulted alignments of individual methods are combined using specific operators, e.g. by taking the union of results. Given a set of k alignment methods (e.g. string-based, WordNet-based), each method computes different confidence values concerning any assessed relation (E_1, E_2, r). The synthesis of these k methods aims to compute an alignment of the input ontologies, with respect to the confidence values of the individual methods. Trimming of the resulted correspondences in terms of a threshold confidence value is also performed for optimization.

The alignment algorithm followed in this work is outlined in the following steps:

– Step 0: If non-English names of labels of entities are detected, translate input ontology into an English-language copy of it.

- Step 1: For each integrated alignment method *k* compute correspondence (E_i^1, E_j^2, *r*, γ) between elements of the two domain ontologies.
- Step 3: Apply trimming process by allowing agents to change a variable threshold value (of γ) for each alignments set S_k or for the alignments of a synthesized method
- Step 4: Apply synthesis of methods at different levels (currently using union aggregation operator) to the resulted set of alignments S_k .

The proposed ontology alignment approach considers most of the challenges in ontology alignment research [3, 5]. Consider two alignment methods (Figure 1), *m* and *m'*, also called matchers, that are selected based on a fixed synthesis configuration method and used for aligning two input ontologies *o* and *o´*. In case of translation needed, this is performed before entering *m* and *m´* respectively. The resulting alignments are aggregated/merged in *a*, using an aggregation operator (union is the current one used), resulting in another alignment *A´´´* which will be improved by another alignment method *m''* resulting to the final alignment *A´´´´*.

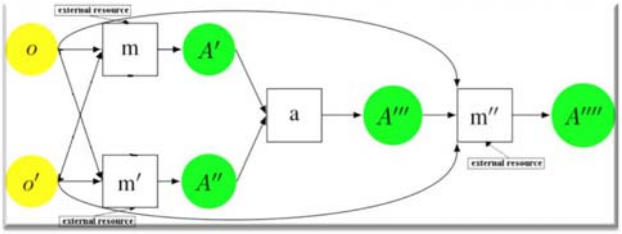


Fig. 1. General description of the ontology alignment process [5]

1.2 Specific techniques used

The tool has been developed by re-using AUTOMSv2 and Alignment API methods and libraries. Specifically, ASE synthesis configuration method merges the alignments of four synthesized alignment methods as described in the following paragraphs, having the first two dedicated to the computation of equivalences and the last two for the computation of subsumptions between ontology entities.

1. *Level 1 (for equivalences)*: Synthesis of three string-based similarity methods, one for each type of entity information i.e. names, labels and comments. For names similarity we use "smoaDistance" from Alignment API, for labels and comments similarity we use COCLU-based methods from AUTOMSv2. For each method a different threshold value is set (0.987 for COCLU-based and 0.82 for SMOA).
2. *Level 2 (for equivalences)*: Synthesis of two WordNet-based similarity methods for discovering synonyms between concept/property pairs, one for each type of entity information i.e. names and labels. For names similarity we use “basicSynonymySimilarity” from Alignment API and for labels we use our own method that is however based on the same basic synonym similarity approach.

3. *Level 3 (for subsumptions)*: Synthesis of two WordNet-based similarity methods for discovering subsumption relations between concept/properties, one for each direction i.e. a>b and a<b. We have developed these custom in-house methods only for labels, and totally depended on WordNet. So, if a hyperonym or hyponym relation between two terms exist in WordNet lexicon, then we conclude a subsumption axiom between the related ontology classes/properties.
4. *Level 4 (for subsumptions)*: Synthesis of two string-based similarity methods for discovering subsumption relations between concept/properties, one for each direction i.e. a>b and a<b. We have developed these custom in-house methods only for labels, and totally depended on the heuristic of compound terms such as: if there is a compound term (e.g. shortName) such as the right-most part of it can be matched to a non-compound term (e.g. name), then we can introduce a subsumption relation between these two such as the compound term is more specific than the non-compound e.g. shortName < Name (i.e. a short name is a kind of name).

The String Matching for Ontology Alignment (SMOA) method utilizes a specialized string metric "smoaDistance" for ontology alignment, first published in ISWC 2005 conference [6].

The WordNet-based string-based similarity distance 'basicSynonymySimilarity' computes the similarity of two terms based in their synonymic similarity, i.e. if they are synonyms in WordNet lexicon (returns '1' if term-2 is a synonym of term-1, else returns a BasicStringDistance similarity score between term-1 and term-2).

The state-of-the-art string similarity distance method COCLU, initially integrated in AUTOMS [4] and in other implementations using the AUTOMS-F API [7] is a partition-based clustering algorithm which divides data into clusters and searches the space of possible clusters using a greedy heuristic.ASE completely re-implements it and uses it in two different modes, i.e. in labels-mode and in comment-mode.

The large dependency of our alignment methods in an external resource such as WordNet is due to the specific requirement of the application domain that ASE is used in i.e. ontologies are very often flat (no structure), have no instances (unpopulated), have very few concepts/properties (1 to 5 in most cases), have no expressive axioms and compound terms are very common.

1.3 Link to the system and to the set of provided alignments (in align format)

ASE web page (short description, the system and OAEI results) is currently hosted at http://ai-lab-webserver.aegean.gr/kotis/ASE.

2 Results

The results reported in OAEI 2012 contest has been computed with an ASE version that does not integrate the methods for discovering subsumption relations between entities. This was decided due to the nature of the 'refaligns' provided by some organizers for some datasets. For instance, in Benchmark track, although a

meaningful alignment between shortName and Name should have been included in the reference alignments with a subsumption relation (a ShortName is a Name), this was not the case. So, in order to avoid low precision due to this matter, we decided to exclude the capability of computing subsumption alignments for all tests.

2.1 Benchmark 2012

The Benchmark results for OAEI 2012 (http://oaei.ontologymatching.org/2012/benchmarks/index.html) indicated that ASE could not perform high in terms of precision (ranging between 0.27 and 0.72) but stay at the same levels as our AUTOMSv2 in terms of recall (ranging between 0.51 and 0.54) for the four out of five domains (see Table 1). For the last domain, i.e. finance (blind test), the tool did not compute any results. The low precision results however were related to additional mappings that have been recorded in the output alignment string, computed by one third-party method we reused (smoaDistance in Alignment API) which also aligns instances that are found in the ontologies (aligned entities can be classes, properties, and instances). At the same time, the reference alignments of Benchmark do not contain mappings of instances.

Having said that, since it is based in AUTOMSv2 alignment methods and Alignment API framework, we can expect that the corrected version will approximate at least the precision scores of AUTOMSv2 for this track (since AUTOMSv2 is the baseline for ASE development). This issue can be also supported by the fact that ASE computes the higher precision (0.72) for those datasets that have no (or the less) instances of all datasets i.e. benchmark-2.

Table 1. Scores for Benchmark track 2012

	Precision	F-measure	Recall	Runtime(s):
biblio	0.49	0.51	0.54	26
benchmark-2	0.72	0.61	0.53	69
benchmark-3	0.27	0.36	0.54	690
benchmark-4	0.4	0.45	0.51	276
finance	n/a	n/a	n/a	n/a

2.2 Conference 2012

The Conference results for OAEI 2012 (http://oaei.ontologymatching.org/2012/conference/index.html) indicated that ASE could perform higher in terms of precision (range between 0.61 and 0.63) and lower for recall (range between 0.4 and 0.43).

ASE failed to generate 6 alignments out of 120 testcases. Improved version delivered after deadline succeeded to generate all alignments (with improved scores, as in AUTOMSv2) however because it was delivered after deadline (and precision and recall performance was different) official results are reported according to initial submitted version. Runtime is reported according to the latest version which does not differ with the initial version much.

In this paper we decided to present (see Table 2), only the results generated with the official version of our tool (before the deadline of the contest), and not the one generated with an improved version (fixing unexpected third-party library crash) submitted after the deadline. This decision was made due to the feedback that we received from organizers of this track.

Table 2. Scores for Conference track 2012

Official (before deadline)				
	Precision	**F-measure**	**Recall**	**Runtime(ms)**
r1	0.63	0.51	0.43	104371
r2	0.61	0.48	0.4	104371

Comparing to AUTOMSv2 results for this track, ASE has generally an improved performance (f-measure is higher for both subtests), based mainly on the higher recall scores that we obtained. Also, runtime is quite improved (almost ¼ of AUTOMSv2 runtime).

Finally, we argue that if ASE was running on its full version, i.e. integrating also the methods for discovering subsumption relations between entities, it would have been achieved higher scores (sacrificing however performance in terms of runtime).

2.3 MultiFarm 2012

ASE was not able to compute official Multifarm results for OAEI 2012 (http://www.irit.fr/OAEI/). That was due to an unexpected crash of our third-party on-line translation API (Bing Translator) at the time of ASE execution by organizers.

Although we have immediately replaced this library with the one we use in AUTOMSv2, produced results for OAEI 2011.5 and OAEI 2012 campaigns, and obtained results also with ASE for this dataset, we do not report them here. In this paper we decided to present results generated with the official version of our tool (before the deadline of the contest) and not the ones generated with an improved version (fixing unexpected third-party library crash) submitted after the deadline. That decision was made due to the feedback and recommendation that we received from organizers of this track.

Table 3. Scores for MultiFarm track 2012

Unofficial results (after deadline)			
	Runtime	**Precision**	**Recall**
Lower	2687	0.15	0.00
Higher	237971	0.93	0.57
Average	18570	0.63	0.31

Having said that, from the results we obtained with the fixed unofficial version, we were able to gather good results (ranging between 0.15 and 0.93 for precision, 0 and 0.57 for recall, with largest runtime 237971s, and averages for precision=0.63,

recall=0.31 and runtime=18570s), results that could be easily compared to AUTOMSv2 results for this track.

3 Comments

As already stated, the aim of this development experience, as with our baseline tool AUTOMSv2, was not to develop a tool to compete with others in terms of precision and recall. Instead, we aimed at the development of a subversion of AUTOMSv2 in order to fit in our application domain of IoT. Nevertheless, ASE obtained some good results (although not with the official OAEI 2012 version). As a general comment, ASE sacrificed precision (not much of recall though) for speed, since it uses only a subset of the alignment methods implemented in AUTOMSv2.

The following table summarizes the features of ASE tool:

Num. of input ontologies:	2
Ontology Elements:	Classes, Properties, Instances
Mapping cardinality:	1:1
Formal Language:	OWL
Relation:	=, <, >
Confidence:	[0, 1]
Natural Language:	EN, DE, FR, NL, ES, PT

ASE results could have been better (if using the latest unofficial version that we submitted after the deadline) and computation of results could have been performed also for other tracks (Library, Anatomy, LargeBio). We experienced a lot of unexpected difficulties with bugs appeared last minute in third-party libraries such as in Alignment API, COCLU string similarity method, WebTranslator API, and Microsoft Bing Translator API.

ASE is participating in this contest with its first prototype version. We plan to optimize its performance by testing and adapting new configurations of synthesized methods in a more efficient manner, always having AUTOMSv2 as our baseline tool.

In our future plans it is also the creation of a custom dataset and reference alignments using ontologies for the specific domain of IoT and Smart Environments. This is needed in order to better explore the requirements of such domain-specific evaluation of an ontology alignment tool. As it has been already stated, ASE must be evaluated in its context i.e. using ontologies that are very often flat (no structure), have no instances (unpopulated), have very few concepts/properties (1 to 5 in most cases), have no expressive axioms and compound terms are very common.

4 Conclusion

This paper presented ASE tool and official evaluation results obtained for OAEI 2012 contest. The effort was based on experience gained by the development of

AUTOMSv2 for OAEI 2011.5 and OAEI 2012. The development process of this tool was driven by our motivation to use the ontology alignment functionality as part of the Smart Proxy approach for the matchmaking of Internet of Things entities. In this paper we decided to present results generated with the official version of our tool (before the deadline of the contest) and not the ones (better in some cases) generated with the improved version (fixing unexpected third-party library crashes) submitted after the deadline. That decision was made due to the feedback and recommendation that we received from organizers of this track.

Acknowledgements

We thank all organizers for the valuable feedback and assistance towards delivering the presented results. We also acknowledge the work of developers/researchers in AUTOMS, AUTOMS-F and SMOA.

References

1. David, J., Euzenat, J., Scharffe, F., Trojahn dos Santos, C.: The Alignment API 4.0, Semantic Web - Interoperability, Usability, Applicability, 2(1):3-10, IOS Press (2011)
2. Euzenat, J., Meilicke, C., Stuckenschmidt, H., Shvaiko, P., Trojahn, C.: Ontology Alignment Evaluation Initiative: six years of experience, J. Data Semantics 15: 158-192 (2011)
3. Kotis, K., Lanzenberger, M.: Ontology Matching: Current Status, Dilemmas and Future Challenges. In: International Conference of Complex, Intelligent and Software Intensive Systems, pp. 924-927 (2008)
4. Kotis, K., Valarakos, A., Vouros, G. A.: AUTOMS: Automating Ontology Mapping through Synthesis of Methods, In: International Semantic Web Conference, Ontology Matching International Workshop, Atlanta USA (2006)
5. Shvaiko, P., Euzenat, J.: Ontology matching: state of the art and future challenges, IEEE Transactions on Knowledge and Data Engineering, 08 Dec. 2011. IEEE computer Society Digital Library. IEEE Computer Society, http://doi.ieeecomputersociety.org/10.1109/TKDE.2011.253
6. Stoilos, G., Stamou, G., Kollias, S.: A String Metric for Ontology Alignment. In: International Semantic Web Conference (2005)
7. Valarakos, A., Spiliopoulos, V., Kotis K., Vouros, G. A.: AUTOMS-F: A Java Framework for Synthesizing Ontology Mapping Methods, In: International Conference i-Know, Graz, Austria (2007)

AUTOMSv2 Results for OAEI 2012

Konstantinos Kotis[1], Artem Katasonov[1], Jarkko Leino[1]

[1] VTT Technical Research Centre of Finland, Tampere, FI
{Ext-konstantinos.kotis, Artem.Katasonov, Jarkko.Leino}@vtt.fi

Abstract. This paper presents AUTOMSv2 effort towards building a tool for the automated alignment of domain ontologies. The developed tool is a result of our motivation to rebuild AUTOMS tool (presented in OAEI 2006) by putting together a) a well-known, widely used and continuously evolving/maintained alignment framework b) the synthesis of state-of-the-art alignment methods, c) a modern approach of synthesizing methods using profiling and configuration strategies, and d) multilingual support. The aim of this experience was not to compete with other tools in precision and recall but to re-develop AUTOMS using the abovementioned technologies and methods. Nevertheless, AUTOMSv2 obtained satisfactory results when compared with tools of OAEI 2011 and 2011.5 campaigns.

1 Presentation of the system

1.1 State, purpose, general statement

AUTOMSv2 is an automated ontology alignment tool based on its early version (AUTOMS) in 2006 [4]. It computes 1:1 (one to one) alignments of two input domain ontologies in OWL, discovering equivalences between ontology elements, both classes and properties. The features that this new version integrates are summarized in the following points:

- It is implemented with the widely used open source Java Alignment API [1]
- It synthesizes alignment methods at various levels and types (lexical, structural, instance-based, vector-based, lexicon-based) with the capability to aggregate their alignments using different aggregation operators (union, Pythagorean means)
- It implements an alignment-methods' configuration strategy based on ontology profiling information (size, features, etc.)
- It integrates state-of-the-art alignment methods with standard Alignment API methods
- Implements a language translation method for non-English ontology elements

This work was carried out during the tenure of an ERCIM "Alain Bensoussan" Fellowship Programme. This Programme is supported by the Marie Curie Co-funding of Regional, National and International Programmes (COFUND) of the European Commission

The problem of computing alignments between ontologies can be formally described as follows: Given two ontologies $O_1 = (S_1, A_1)$, $O_2 = (S_2, A_2)$ (where S_i denotes the signature and A_i the set of axioms that specify the intended meaning of terms in S_i) and an element (class or property) E_i^1 in the signature S_1 of O_1, locate a corresponding element E_j^2 in S_2, such that a mapping relation (E_i^1, E_j^2, r) holds between them. r can be any relation such as the equivalence (≡) or the subsumption (⊑) axiom or any other semantic relation e.g. meronym. For any such correspondence a mapping method may relate a value γ that represents the preference to relating E_i^1 with E_j^2 via r. If there is not such a preference, we assume that the method equally prefers any such assessed relation for the element E_1. The correspondence is denoted by (E_i^1, E_j^2, r, γ). The set of computed mapping relations produces the mapping function $f:S_1 \rightarrow S_2$ that must preserve the semantics of representation: i.e. all models of axioms A_2 must be models of the translated A_1 axioms: i.e. $A_2 \models f(A_1)$.

The synthesis of alignment methods that exploit different types of information (lexical, structural, and semantic) and may discover different types of relations between elements has been already proved to be of great benefit [2, 5]. Based on the analysis of the characteristics of the input ontology definitions, i.e. the profiling of ontologies, our approach provides different configurations (syntheses) of alignment methods. The analysis of input ontologies is based on their size, the existence of individuals or not, the existence of class/properties annotations e.g. labels, and the existence of entity names with an entry in WordNet lexicon. Part of the profiling is also a translation method that supports the translation of classes/properties annotations if these are given in a non-English language.

In the presented work we follow a modern synthesis strategy, which performs composition of results at different levels (see Figure 1): the resulted alignments of individual methods are combined using specific operators, e.g. by taking the union or intersection of results, intersection of results or by combining the methods' different confidence values with weighing schemas. Given a set of k alignment methods (e.g. string-based, vector-based), each method computes different confidence values concerning any assessed relation (E_1, E_2, r). The synthesis of these k methods aims to compute an alignment of the input ontologies, with respect to the confidence values of the individual methods. Trimming of the resulted correspondences in terms of a threshold confidence value is also performed for optimization.

The alignment algorithm followed in our work is outlined in the following steps:

- Step 1: Analyze ontology definitions to be aligned (profiling step) and assign the correspondent configuration of alignment methods to be used (configuration step). If needed, translate ontology into an English-language copy of it.
- Step 2: For each integrated alignment method k compute correspondence (E_i^1, E_j^2, r, γ) between elements of the two domain ontologies.
- Step 3: Apply trimming process by allowing agents to change a variable threshold value for each alignments set S_k or for the alignments of a synthesized method
- Step 4: Apply synthesis of methods at different levels (currently using union aggregation operator) to the resulted set of alignments S_k .

The proposed ontology alignment approach considers most of the challenges in ontology alignment research [3, 5] but emphasizes the alignment methods selection and synthesis.

1.2 Specific techniques used

The tool has been developed from scratch, reusing some of the alignment methods already provided within the Alignment API. Other state-of-the-art methods such as the COCLU string-based and the LSA vector-based methods implemented in AUTOMS [4] using the AUTOMS-F API [7] have been re-implemented using the new API. The instance-based and structure-based alignment methods have been also implemented from scratch. The detailed description of the alignment methods have been presented already in previously published works [4, 6, 7]. The integrated string-based methods are used in two different synthesized methods and in one single method. All three methods use class and property names as input to their similarity distance metrics.

The first synthesized method, synthesizes the alignments of two string-based similarity distance methods distributed with the Alignment API, namely, the 'smoaDistance' method and the 'levenshteinDistance'. A general Levenshtein distance between two strings is defined as the minimum number of edits needed to transform one string into the other, with the allowable edit operations being insertion, deletion, or substitution of a single character. The one re-used from the Alignment API is a version of the general distance metric, based on the Needleman Wunsch distance method. The String Matching for Ontology Alignment (SMOA) method utilizes a specialized string metric for ontology alignment, first published in ISWC 2005 conference [6].

The second synthesized method, synthesizes the alignments of two WordNet-based string-based similarity distance methods of the Alignment API, namely, the 'basicSynonymySimilarity' and the 'cosynonymySimilarity'. The first computes the similarity of two terms based in their synonymic similarity, i.e. if they are synonyms in WordNet lexicon (returns '1' if term-2 is a synonym of term-1, else returns a BasicStringDistance similarity score between term-1 and term-2), and the second computes the proportion of common *synsets* between them, i.e. the proportion of common synonyms shared by both terms.

The third one is a single method that is implemented based on the state-of-the-art string similarity distance method COCLU, initially integrated in AUTOMS [4] and in other implementations using the AUTOMS-F API [7]. Since AUTOMSv2 completely re-implements it, it is used in two different modes, i.e. in names-mode and in labels-mode, according to the type of input ontologies that the profiling method will return. COCLU is a partition-based clustering algorithm which divides data into clusters and searches the space of possible clusters using a greedy heuristic.

Regarding vector-based alignment methods, AUTOMSv2 integrates two LSA-based methods, versions of the original HCONE-merge alignment method implemented in AUTOMS [4]. The first version is based on LSA (Latent Semantic Analysis) and WordNet and the second just in LSA. In the first one, given two ontologies, the algorithm computes a morphism between each of these two ontologies

and a “hidden intermediate” ontology. This morphism is computed by the Latent Semantic Indexing (LSI) technique and associates ontology concepts with WordNet senses. Latent Semantic Indexing (LSI) is a vector space technique originally proposed for information retrieval and indexing. It assumes that there is an underlying latent semantic space that it estimates by means of statistical techniques using an association matrix (n×m) of term-document data (WordNet senses in this case). The second version of this method is based on the same idea but instead of exploiting WordNet senses it builds the term-document matrix from the concepts' names/labels/comments and their vicinity (properties, direct super-concepts, direct subconcepts) of the input ontologies. The similarity between two vectors (each corresponding to class name and annotation as well as to its vicinity) is computed by means of the cosine similarity measure.

Finally, two more methods, a structure-based and an instance-based method, are integrated, based on the general principle that two classes can be considered similar if a percentage of their properties or their instances has been already considered to be similar. The similarity of properties and instances is computed using a simple string-matching method (Levenshtein). In case structure and instances are not common in the input ontologies, their integration in AUTOMSv2 does not influence its performance since, as already stated, the profiling analysis automatically detects the features of the input ontologies and exclude these methods from computing alignments (i.e. are not included in the synthesis configuration for the smart/control entities' ontology definitions).

The different configurations regarding the way the above methods were synthesized, i.e. computing and synthesizing alignments, is based on the profiling information gathered after the analysis of the input ontologies. Both input ontologies (since our problem concerns the alignment of two ontologies), are examined using different analysis methods, as the example following ones:

1. Based on the size of the ontologies, i.e. the number of classes that ontologies have, if one of them has more than a specific number of classes (this number is experimentally set to 100), then this pair of ontologies is not provided as input to alignment methods with heavy computations since it will compromise the overall execution time of the tool. Such methods are the vector-based, WordNet-based and structure-based ones.
2. If an ontology pair contains an ontology with no instances at all, then this pair is not provided as input to any instance-based alignment method (the explanation for this is straight forward).
3. If an ontology pair contains two ontologies that a specific number of their entities have no names with an entry in WordNet, but they have labels, then provide this pair as input to alignment methods that a) do not consider WordNet as an external resource and b) consider labels matching instead of class names.
4. If an ontology pair contains two ontologies that a specific number of their entities have no names with an entry in WordNet, and they also have no labels, then provide this pair as input to alignment methods that a) do not consider WordNet as an external resource and b) do not consider labels' matching.

AUTOMSv2 is using a free Java API named WebTranslator (http://webtranslator.sourceforge.net/) in order to solve the multi-language problem. AUTOMSv2 translation method is converting the labels of classes and properties that are found to be in a non-English language (any language that WebTranslator supports) and creates a copy of an English-labeled ontology file for each non-English ontology. This process is performed before AUTOMSv2 profiling, configuration and matching methods are executed, so their input will consider only English-labeled copies of ontologies.

1.3 Link to the system and to the set of provided alignments (in align format)

AUTOMSv2 web page (short description, the system and OAEI results) is currently hosted at http://ai-lab-webserver.aegean.gr/kotis/AUTOMSv2.

2 Results

In this paper we conjecture that we must also shortly present a snapshot of AUTOMSv2 participation in 2011.5 campaign. This was motivated by the capability of giving a rough comparison with other tools also participated in the same contest, and also comparing it with latest versions of our own tools that participated in the OAEI 2012 contest. A pre-final experimental version of AUTOMSv2 was submitted in 18th of March 2012 as a submission to the Ontology Alignment Evaluation Initiative 2011.5 Campaign (http://oaei.ontologymatching.org/2011.5/seals-eval.html), using the Semantic Evaluation At Large Scale (SEALS) platform.

The Benchmark results ("biblio" dataset) for OAEI 2011.5 (http://oaei.ontologymatching.org/2011.5/results/benchmarks/index.html) indicated that AUTOMSv2 could perform quite high in terms of precision (0.97) and low for recall (0.54). Its f-measure (0.69) was the 6th best in 14 tools participated (only for this particular dataset). In terms of runtime measurements, AUTOMSv2 was placed in the 8th place in 13 tools, which was not an expecting result due to the profiling and configuration optimization strategy the AUTOMSv2 follows.

The Conference results for OAEI 2011.5 (http://oaei.ontologymatching.org/2011.5/results/conference/index.html) again indicated that AUTOMSv2 could perform quite higher in terms of precision (0.75 and 0.79) and lower for recall (0.4 and 0.43), where the highest precision of other tools was 0.78 and 0.82. In terms of runtime performance AUTOMSv2 performed quite similar to Benchmark results.

The Multifarm results for OAEI 2011.5 (http://oaei.ontologymatching.org/2011.5/results/multifarm/index.html) indicated that AUTOMSv2 could perform quite well with multilingual ontologies, obtained the 2nd better f-measure result (0.36) among 12 tools (for type I dataset – different ontologies), with an average precision of 0.63 and a recall of 0.25.

For Anatomy and Large Biomedical Ontologies tracks of OAEI 2011.5, AUTOMSv2 did not generate any results.

2.1 Benchmark 2012

The Benchmark results for OAEI 2012 (http://oaei.ontologymatching.org/2012/benchmarks/index.html) indicated that AUTOMSv2 could perform quite high in terms of precision (range between 0.91 and 0.99) and low for recall (range between 0.51 and 0.55) for the four out of five domains (see Table 1). For the last domain, i.e. finance, the tool performed similarly in terms of recall (0.55) but unexpectedly (blind test) in terms of precision (0.35). Comparing to 2011.5 results, AUTOMSv2 has not improved its performance.

Table 1. Scores for Benchmark track 2012

	Precision	F-measure	Recall	Runtime(s):
biblio	0.97	0.69	0.54	58
benchmark-2	0.97	0.68	0.52	161
benchmark-3	0.99	0.7	0.54	519
benchmark-4	0.91	0.65	0.51	421
finance	0.35	0.42	0.55	1535

2.2 Conference 2012

The Conference results for OAEI 2012 (http://oaei.ontologymatching.org/2012/conference/index.html) indicated that AUTOMSv2 could perform higher in terms of precision (range between 0.64 and 0.67) and lower for recall (range between 0.33 and 0.36).

AUTOMSv2 failed to generate 6 alignments out of 120 test cases. An improved version delivered after deadline succeeded to generate all alignments however because it was delivered after deadline (and precision and recall performance was different) official results are reported according to initial submitted version. Runtime is reported according to the latest version which does not differ with the initial version much. Having said that, improved version delivered after deadline succeeded to generate all alignments with improved performance (in the case of ra1: Precision=0.79, F1-measure=0.56, Recall=0.43 and in the case of ra2: Precision=0.75, F1-measure=0.52, Recall=0.4)

Table 2. Scores for Conference track 2012

Official (before deadline)				
	Precision	F-measure	Recall	Runtime(ms)
r1	0.67	0.47	0.36	452477
r2	0.64	0.44	0.33	452477
Improved (after deadline)				
	Precision	F-Measure	Recall	Runtime
r1	0.79	0.56	0.43	same
r2	0.75	0.52	0.4	same

In this paper we decided to present (see Table 2), only the results generated with the official version of our tool (before the deadline of the contest) and not the one generated with an improved version (fixing unexpected third-party library crash)

submitted after the deadline. This decision was made due to the feedback that we received from organizers of this track.

Comparing to 2011.5 results, AUTOMSv2 has not improved its performance (compared with the official results).

2.3 Multifarm 2012

The Multifarm results for OAEI 2012 (http://www.irit.fr/OAEI/) indicated that AUTOMSv2 could perform for all pairs apart from the ones involving Czech, Russian and Chinese.

Table 3. Scores for Multifarm track 2012

Official (before deadline)				
	Precision	**F-measure**	**Recall**	**Runtime(s)**
de-en	0.91	0.35	0.22	891
de-es	0.82	0.26	0.15	1752
de-fr	0.93	0.25	0.14	1842
de-nl	0.88	0.31	0.19	1694
de-pt	0.9	0.25	0.15	1714
en-es	0.71	0.32	0.21	886
en-fr	0.75	0.32	0.2	1006
en-nl	0.78	0.35	0.23	851
en-pt	0.75	0.29	0.18	926
es-fr	0.74	0.29	0.18	1668
es-nl	0.7	0.34	0.22	1757
es-pt	0.7	0.36	0.25	1748
fr-nl	0.71	0.26	0.16	1735
fr-pt	0.74	0.26	0.16	1699
Average	0.79	0.30	0.19	1441

For the non-zero computed pairs, the tool performed higher in terms of precision (range between 0.7 and 0.91) and lower for recall (range between 0.14 and 0.25). In this paper we decided to present results (see Table 3) generated with the official version of our tool (before the deadline of the contest) and not the ones generated with an improved version (fixing unexpected third-party library crash) submitted after the deadline. That decision was made due to the feedback that we received from organizers of this track also.

Comparing to 2011.5 results, AUTOMSv2 has not improved its performance. In fact, the f-measure has been decreased by 0.6. Comparing the average results of precision and recall between the two contests, we can observe that the average precision was increased while the average recall was decreased significantly.

2.4 LargeBio 2012

The LargeBio results for OAEI 2012 indicated that AUTOMSv2 could perform also with large datasets, although with large runtimes (17 hours). The results are depicted in Table 4. As expected, AUTOMSv2 could perform higher in terms of precision (range between 0.79 and 0.82) and lower for recall (range between 0.49 and 0.52).

Table 4. Scores for LargeBio track 2012

FMA-NCI	Precision	Recall
Original UMLS mappings	0.82	0.49
Refined UMLS mappings using LogMap's repair facility	0.80	0.50
Refined UMLS mappings using Alcomo debugging system	0.79	0.51
Harmonized mapping set from OAEI 2011.5	0.82	0.52

3 Comments

As already stated, the aim of this development experience was not to deliver a tool to compete with others in terms of precision and recall. Instead, we aimed at the development of a new version of AUTOMS (Automating the Synthesis of Ontology Mapping Methods) using new and state-of-the-art technologies and alignment methods. Nevertheless, AUTOMSv2 obtained good (above average) results both in OAEI 2011.5 and 2012 contests.

The following table summarizes the features of ASE tool:

Num. of input ontologies:	2
Ontology Elements:	Classes, Properties
Mapping cardinality:	1:1
Formal Language:	OWL
Relation:	=
Confidence:	[0, 1]
Natural Language:	EN, DE, FR, NL, ES, PT

AUTOMSv2 results could have been better and computation of results could have been performed for other tracks (Library, Anatomy). We experienced a lot of unexpected difficulties with bugs appeared in third-party libraries such as in Alignment API, COCLU string similarity method, WebTranslator API, Microsoft Bing Translator API.

Our future plans to integrate also the computation of subsumption relation between concepts/properties has been lately realized in a new tool called ASE (Aligning Smart Entities), also participating in this contest as a first prototype version. Also, we plan to optimize the performance of our ontology alignment tools by adapting the configurations of the synthesized methods in a more efficient manner.

4 Conclusion

This paper presented AUTOMSv2 tool and evaluation results obtained for OAEI 2011.5 and 2012 contests. This effort was the result of our motivation to rebuild AUTOMS by putting together a) a well-known, widely used and continuously evolving/maintained alignment framework b) the synthesis of state-of-the-art

alignment methods, c) a modern approach of synthesizing methods using profiling and configuration strategies, and d) multilingual support. Although our aim was not to compete with other tools in precision and recall, nevertheless, AUTOMSv2 obtained good results that we have also compared with results of other tools obtained for OAEI 2011 and 2011.5 contests.

References

1. David, J., Euzenat, J., Scharffe, F., Trojahn dos Santos, C.: The Alignment API 4.0, Semantic Web - Interoperability, Usability, Applicability, 2(1):3-10, IOS Press (2011)
2. Euzenat, J., Meilicke, C., Stuckenschmidt, H., Shvaiko, P., Trojahn, C.: Ontology Alignment Evaluation Initiative: six years of experience, J. Data Semantics 15: 158-192 (2011)
3. Kotis, K., Lanzenberger, M.: Ontology Matching: Current Status, Dilemmas and Future Challenges. In: International Conference of Complex, Intelligent and Software Intensive Systems, pp. 924-927 (2008)
4. Kotis, K., Valarakos, A., Vouros, G. A.: AUTOMS: Automating Ontology Mapping through Synthesis of Methods, In: International Semantic Web Conference, Ontology Matching International Workshop, Atlanta USA (2006)
5. Shvaiko, P., Euzenat, J.: Ontology matching: state of the art and future challenges, IEEE Transactions on Knowledge and Data Engineering, 08 Dec. 2011. IEEE computer Society Digital Library. IEEE Computer Society, http://doi.ieeecomputersociety.org/10.1109/TKDE.2011.253
6. Stoilos, G., Stamou, G., Kollias, S.: A String Metric for Ontology Alignment. In: International Semantic Web Conference (2005)
7. Valarakos, A., Spiliopoulos, V., Kotis K., Vouros, G. A.: AUTOMS-F: A Java Framework for Synthesizing Ontology Mapping Methods, In: International Conference i-Know, Graz, Austria (2007)

GOMMA Results for OAEI 2012

Anika Groß, Michael Hartung, Toralf Kirsten, and Erhard Rahm

Department of Computer Science, University of Leipzig, Germany
{gross,hartung,tkirsten,rahm}@informatik.uni-leipzig.de

Abstract. We present the OAEI 2012 evaluation results for the matching system GOMMA developed at the University of Leipzig. The original application focus of GOMMA has been the life science domain but as a generic tool it can also match ontologies from other areas. It could thus participate in all OAEI tracks running on the SEALS platform. GOMMA supports several methods for efficiently matching large ontologies in particular parallel matching on multiple cores or machines, reducing the search space as well as reusing and composing previous mappings to related ontologies.

1 Presentation of the system

1.1 State, purpose, general statement

GOMMA (**G**eneric **O**ntology **M**atching and Mapping **Ma**nagement) [6] is a comprehensive infrastructure to manage and analyze the evolution of life science ontologies and mappings [4]. It includes a generic component to semantically align (match) ontologies. GOMMA is able to match very large ontologies as common in the life sciences. To deal with large ontologies GOMMA provides several scalable match techniques:

1. Parallel ontology matching on multiple computing nodes and CPU cores [2],
2. Indirect computation of ontology mappings by reusing and composing previously determined ontology mappings via intermediate ontologies [3], and
3. A newly introduced blocking approach to reduce the search space by restricting matching to overlapping ontology parts.

These techniques all support efficiency, in particular reduced computation times. The latter two approaches can also improve match quality. While the original focus of GOMMA has been in the life science domain, the match component is generic. We could thus participate in all 2012 match problems of the Ontology Alignment Evaluation Initiative (OAEI)[1] running on the SEALS platform.

1.2 GOMMA Matching Workflow

The GOMMA matching workflow used for OAEI 2012 is displayed in Fig. 1. In the following, we describe its three main phases, namely the initial phase (including the new blocking strategy), the matching phase as well as a set of postprocessing steps.

[1] http://oaei.ontologymatching.org

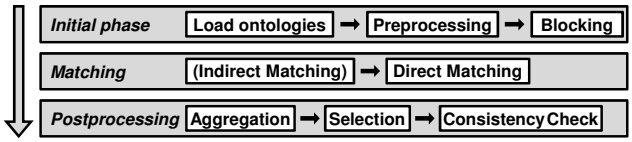


Fig. 1. GOMMA matching workflow for OAEI 2012

Generally, the input of the matching are two ontologies, source O_1 and target O_2, each consisting of concepts (classes, properties) as well as a structure (relationships between concepts, e.g., is_a, part_of). Internally, ontologies are represented as rooted, acyclic graphs. A concept has different attributes such as its name or a set of synonyms. The output of the matching workflow is a mapping M consisting of a set of correspondences whereby each correspondence has a similarity value denoting the strength of the connection between two concepts c_1 and c_2: $M = \{(c_1, c_2, sim) \,|c1 \in O_1, c2 \in O_2\}$.

Initial Phase and Blocking In the initial phase we first parse and *load the ontologies*. In this step, we assign all information relevant for matching to concepts, in particular name, synonyms, comments and instances. Note, that some attributes are multi-valued, e.g., there can be several synonyms or instances per concept. The information is stored within text attributes and used for string-based match comparisons.

During *preprocessing* we also check the language of attribute values (using xml:lang of rdfs:label). In case it is different from English we translate the term to English and add it as a new synonym to the concept. We used a free translation API[2] to automatically translate non-English terms. Using this facility, we iteratively established a dictionary to store the retrieved synonyms. All concept attribute values are further *normalized*, i.e., we remove delimiters and stop words, and normalize strings to lower case.

In the initial phase, we further apply a *blocking* strategy to reduce the number of comparisons for large ontologies. There have been various approaches to reduce the search space for large scale matching (see [7] for a recent survey). Our current approach is different and focuses on "asymmetric" match problems where a specific ontology is matched to a broader ontology from which only a part is relevant. An example for such an asymmetric match problem is the alignment of a pure anatomy ontology such as the Foundational Model of Anatomy (FMA) against a broad biomedical ontology such as NCI Thesaurus covering anatomy in one part. Another scenario for linked data is to match a domain-specific ontology, e.g. from the geographical domain, to the broad DBpedia ontology.

To deal with such match problems we aim at automatically identifying the relevant part of the broader ontology and to match only this part with the more specific, and typically smaller ontology. This blocking strategy is expected to (1) dramatically improve efficiency in applicable cases and (2) improve match quality (in particular precision) due to fewer false positive correspondences. The blocking strategy is based on an initial mapping and works in the following steps:

[2] http://mymemory.translated.net/

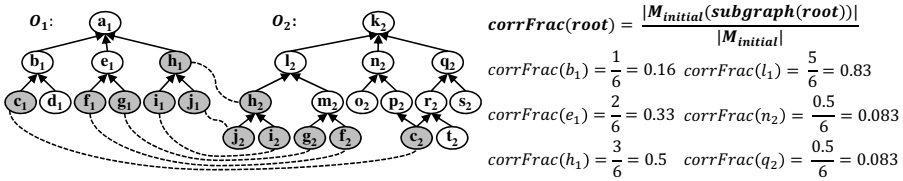


Fig. 2. Blocking ontology subgraphs

1. Determine an initial mapping $M_{initial}$ using a very efficient match method, e.g., exact name matching with hashed attribute values (applied in OAEI 2012) or the reuse of precomputed mappings.
2. Identify a set of subgraph *roots* below the top root. Determine the number of correspondences from $M_{initial}$ per subgraph root, $|M_{initial}\,(subgraph\,(root))|$, by propagating the correspondence counts from the leaf level upwards. In case of multiple inheritance, the correspondence count is partially propagated upwards the ontology structure (for the example in Fig. 2 this is done for O_2 concept c_2).
3. For each $root$ compute a correspondence fraction $corrFrac(root)$ that is the number of correspondences assigned to the root $|M_{initial}\,(subgraph\,(root))|$ divided by the overall size of the initial mapping $|M_{initial}|$ (see Fig. 2).
4. Select the most valuable root(s) with a $corrFrac$ above a given threshold. All concepts in the subgraph of this root will be used for matching, other concepts will not be compared. If no root exceeds the threshold, blocking is not applied, i.e., the whole ontology needs to be matched since no dominating part is found.

Fig. 2 illustrates the approach for two ontologies and a set of predetermined correspondences. To choose a promising subgraph for matching, we consider roots on the second ontology level (b_1,e_1,h_1 for O_1 and l_2,n_2,p_2 for O_2). Applying a $corrFrac$ threshold of 0.7 means that a subgraph must cover at least 70% of all initial correspondences. This is only the case for root l_2 in O_2, i.e., in the example only O_2 can be partitioned so that the whole O_1 will be matched with the l_2-subgraph of O_2.

Matching GOMMA's matching component allows for *direct* and *indirect matching* of ontologies. Direct match strategies involve internal ontology knowledge like concept-associated or structural information. By contrast, our indirect matching is based on the composition of existing mappings to intermediate (background) ontologies. To efficiently match especially large ontologies, we further parallelize the direct matching process. In the following we describe the match strategies used for OAEI 2012.

To directly match two ontologies we combine up to three different matchers. We always apply a name/synonym matcher that determines the maximal string similarity for the names and multi-valued synonyms per concept pair. In case the necessary information is available, we also apply a comment matcher and instance matcher. GOMMA supports further matchers such as structural matchers [6] but we found them less effective for life science ontologies. We thus did not include them in our default strategy used for all OAEI tasks.

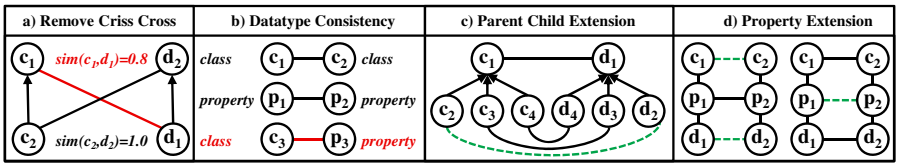


Fig. 3. Consistency checking. Red continuous (green dotted) line = remove (add) correspondence

To efficiently match large ontologies we apply intra-matcher parallelization [2]. For this purpose, we uniformly partition the input ontologies into smaller fragments with the same number of concepts and we solve the fragment-level match tasks in parallel. This parallelization is made easy since for the applied matchers all information used for matching is directly associated to the concepts.

To improve match quality we further apply an indirect composition-based match approach [3]. This approach allows the reuse of existing high quality mappings to efficiently match two so far unmatched ontologies. For example, anatomy ontologies O_1 and O_2 can be matched by composing two mappings O_1 - H and H - O_2 with an intermediate "hub" ontology H, e.g. UMLS. For OAEI we used our direct match strategy to precompute several mappings from the source and target ontology via different intermediate ontologies and combine these composed mappings. Since the resulting mapping may still be incomplete, we identify the unmatched source and target concepts and match them directly to extend the result mapping.

Postprocessing The main task of this phase is the combination or aggregation of the directly and indirectly determined mappings and to select the most likely correspondences from the combined mapping. Before this, we first *filter* out all correspondences per mapping with a similarity below a specified threshold. To combine several mappings we take their union and *average* the similarity values per correspondence. We then apply a *maxDelta selection* [1] for the remaining correspondences. This approach returns for each concept only those correspondences with the maximal similarity value or those within a small delta distance to the maximal value, i.e., we only keep the best correspondences for each source and target concept.

We further apply techniques to improve the consistency of mappings by removing presumably wrong and by adding presumably missing correspondences. We currently check four simple constraints; additional checks may be added in the future to further improve consistency. Fig. 3 shows small exemplary scenarios for each *consistency checker*. The first two conditions check situations that may result in a removal of correspondences (to improve precision), similar as in systems like ASMOV [5]. The two other conditions can lead to the addition of correspondences (to improve recall).

First, correspondences must meet a so-called Criss Cross condition (Fig. 3a), i.e., we eliminate conflicting correspondences (c_1, d_1) and (c_2, d_2) where c_2, is a child of c_1, but d_1 a child of d_2 (or vice versa). One can either remove both correspondences or only remove the one with the lower similarity value. Second, we check the datatype consistency (Fig. 3b). In particular, we remove correspondences between properties and classes, i.e., only class-class / property-property correspondences are allowed.

The first rule to extend the mapping checks whether two concepts match but only a subset of their children (Fig.3c). Here, we add a correspondence for the most similar, unmatched pair of children. Finally, in case of matching properties we add correspondence(s) for the domain/range classes if they have no corresponding class, or we conclude a property match if both, domain and range class, have correspondences (Fig.3d).

1.3 Adaptations made for the evaluation

GOMMA's modular structure helped us to adapt the system to work for the OAEI tasks. One major effort was the adaptation of the ontology import mechanism. We implemented a new SAX-based ontology parser which can be used to load multiple ontologies in parallel via threading. Usually, parallel execution of match workflows in GOMMA requires multiple compute nodes. To better utilize the single machine used for the evaluation, we adapted parallel matching to the use of threading to distribute several match jobs on all available CPU cores on only one machine.

1.4 Link to the system and parameters file

GOMMA is available at `http://dbs.uni-leipzig.de/GOMMA`.

2 Results

We now present and discuss the evaluation results of GOMMA in the OAEI 2012 campaign. We participated in six tracks: Anatomy, Large Biomedical Ontologies, Benchmarks, Library, Conference and Multifarm. Detailed results and descriptions about the used computation environments are provided on the OAEI 2012 result page[3].

2.1 Anatomy and Large Biomedical Ontologies

Anatomy Results For the Anatomy Track two real-world anatomy ontologies namely the Mouse Anatomy (2,744 concepts) and the anatomy part of the NCI Thesaurus (3,304 concepts) should be matched. GOMMA achieves a good F-Measure value of ≈87% in a short amount of time (17 sec.) (Fig.4). In a separate configuration using background knowledge (GOMMA-bk) we apply indirect (composition-based) matching [3] using mappings to three intermediate ontologies (UMLS, Uberon or FMA). By doing so we could increase F-Measure to 92.2% in a reduced execution time (15 sec.).
Large Biomedical Ontologies Results This track was extended w.r.t. its first evaluation in OAEI2011.5. In addition to matching FMA and NCI, two new tasks namely FMA–SNOMED and SNOMED–NCI were introduced. All tasks are divided into three subtasks where *small* and *large* ontology fragments or *whole* ontologies need to be matched. In this track GOMMA's approaches (*composition-based matching*, *parallel matching* and *blocking*) helped to achieve high quality match results with relatively low execution times. Since all ontologies consist of more than 4,000 (and up to 120,000)

[3] OAEI 2012 campaign: http://oaei.ontologymatching.org/2012/results/

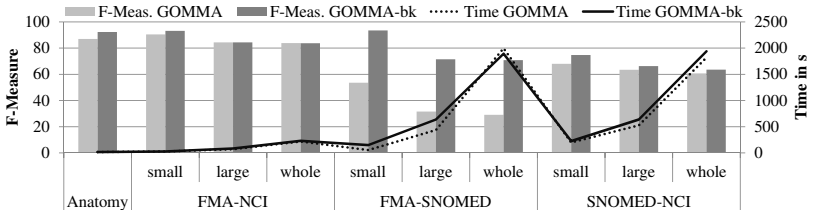


Fig. 4. Evaluation results for the Anatomy and Large Biomedical Ontologies tracks (FMA–NCI, FMA–SNOMED, SNOMED–NCI).

concepts, we apply our blocking strategy (Sec. 1.2) to reduce the overall runtime. Blocking leads to the selection of subgraphs for NCI (FMA–NCI task) and SNOMED (FMA–SNOMED, SNOMED–NCI) thereby reducing the search space by factor 2–6.

The results are summarized in Figure 4. The shown F-Measure values are based on the UMLS reference mapping. There are further results based on refined reference mappings available[4]. As for the Anatomy task, using background knowledge increases the match quality substantially with still acceptable runtime. The best results with 93-94% F-Measure are achieved for the *small* FMA-related subtasks for GOMMA-bk (mappings to UMLS, Uberon). The *small* SNOMED–NCI task seems to be more challenging (≈75% F-Measure with bk). Comparing GOMMA and GOMMA-bk for FMA-SNOMED, we observe a very strong improvement of ≈40% F-Measure when applying *composition-based matching*. For the *whole* FMA-SNOMED (SNOMED-NCI) task we achieve a good F-Measure of 71% (64%) thereby consuming ≈ 30 min computation time. Overall GOMMA-bk takes slightly longer than GOMMA except for the *whole* FMA-SNOMED task. In this case the result of composition-based matching might already cover a higher part of the input ontologies and we do not need to execute a direct matching on whole ontologies.

2.2 Benchmarks and Library

Benchmarks Track Results This track is subdivided into five sub-tracks namely Biblio, Finance and Benchmark 2–4. There are multiple match tasks per sub-track where one source ontology is compared with a number of systematically modified target ontologies. Overall, GOMMA achieved F-Measure values in the range between 60–70% with favoring precision over recall. The recall results are slightly better than in the 2011.5 campaign due to new postprocessors to extend the mapping as described in Sec. 1.2. Using our new thread-based parser, we solved each of the problems in less than one minute.

Library Results In this new, real-world match task the two ontologies STW and TheSoz consisting of about 6,500 and 8,500 concepts need to be aligned. Both ontologies provide a lightweight vocabulary for economic/social science topics and are used in libraries for indexing and search. GOMMA achieved a high recall of ≈91%, however

[4] oaei.ontologymatching.org/2012/results/largeBioMed/

the precision was low (54%). The resulting F-Measure of 67% is comparable to the Benchmark results. Since the vocabularies provide a huge number of labels and synonyms (≈5 per concept), our name/synonym matcher had to evaluate 40,000x32,000 ≈1.3 billion comparisons leading to a runtime of ≈13min. on a 2 core machine.

2.3 Conference and Multifarm

Conference Results The Conference track consists of 16 small ontologies from the domain of conference organization. Each ontology must be matched against each other. In summary, we required about 91 seconds to solve the complete task. The match quality was evaluated against an original (ra1) as well as entailed reference alignment (ra2). For both evaluations we achieved F-Measure values better than the Baseline2 results (61% for ra1 and 56% for ra2). Compared to the 2011.5 campaign, we were able to increase match quality by about 3% in terms of F-Measure (for ra2). In particular, we improved recall by applying the postprocessing methods described in Sec. 1.2.
Multifarm Results The Multifarm task is an extension of the Conference task since conference ontologies in nine different languages (e.g. English, Russian, Chinese) should be matched among each other (36 language pairs). We performed a translation approach (see Sec. 1.2) as a preprocessing step to translate non-English labels into English ones, so that we can afterwards match the translated ontologies with each other. Overall GOMMA required 35 minutes to solve all 36 match problems, i.e. less than one minute per language-pair. The average F-Measure is 35% with an average recall (precision) of 31% (45%). The best results emerge for language pairs where one language is English or for pairs with similar languages, e.g., Spanish to Portuguese with 47% F-Measure.

3 General comments

3.1 Comments on the results and future improvements

The evaluation confirmed that GOMMA has the following strengths:

- Scalable matching of ontologies of different size by performing blocking, parallel matching and mapping composition. A high efficiency and effectiveness is especially achieved in the Anatomy and Large Biomedical Ontologies tracks.
- Substantial improvement of match quality by using domain knowledge, in particular by composing mappings with domain-specific hub ontologies or by applying multi-language translation services for improved synonyms.

We plan to further improve the consistency of the result mapping by applying additional checks during postprocessing. Moreover, we like to apply a more general blocking method to boost both the runtime and match quality (precision) for additional match problems.

3.2 Comments on the OAEI 2012 procedure

Measuring the overall runtimes per match task and system is useful but insufficient to identify and analyze underlying bottlenecks. For example, it would be helpful to see

the time requirements for major phases such as import vs. match. When evaluating scalability (e.g., between a 1-core and a 4-core CPU) the import time might be constant whereas the real match time is reduced with good speed-up. Moreover, it might be interesting to compare the runtime of tools over different years. For each participating tool, available older versions might be re-executed on the currently used machine such that execution times are comparable.

Tools developed by co-organizers of OAEI tracks should not be considered in the official evaluation. This is to avoid the possible suspicion that the design of the match tasks might be tailored to the co-organizers' tools or that the configuration of these tools might be favored by the co-organizers' access to critical data that is unknown for other participants (e.g., Library track gold standard).

4 Conclusion

The participation in six tracks of OAEI 2012 showed that GOMMA is able to efficiently and effectively match ontologies of different size. Especially in the Anatomy and Large Biomedical Ontologies tracks GOMMA's techniques such as *composition-based matching*, *parallel matching* and *blocking* showed to be valuable for a scalable ontology matching. We envision further improvements of GOMMA, e.g. by applying a more general blocking strategy or by additional consistency checks for result mappings.

5 Acknowledgement

Funding: This work is supported by the German Research Foundation (DFG), grant RA 497/18-1 ("Evolution of Ontologies and Mappings").

References

1. Do, H., Rahm, E.: COMA: a system for flexible combination of schema matching approaches. In: Proc. of the 28th Intl. Conf. on Very Large Data Bases (VLDB). pp. 610–621 (2002)
2. Gross, A., Hartung, M., Kirsten, T., Rahm, E.: On matching large life science ontologies in parallel. In: Data Integration in the Life Sciences. pp. 35–49. Springer (2010)
3. Gross, A., Hartung, M., Kirsten, T., Rahm, E.: Mapping composition for matching large life science ontologies. In: Proc. of the 2nd Intl. Conf. on Biomedical Ontology (ICBO), CEUR Workshop Proceedings, CEUR-WS.org/Vol-833/ (2011)
4. Hartung, M., Kirsten, T., Rahm, E.: Analyzing the evolution of life science ontologies and mappings. In: Data Integration in the Life Sciences (DILS). pp. 11–27. Springer (2008)
5. Jean-Mary, Y., Shironoshita, E., Kabuka, M.: Ontology matching with semantic verification. Web Semantics: Science, Services and Agents on the World Wide Web 7(3), 235–251 (2009)
6. Kirsten, T., Gross, A., Hartung, M., Rahm, E.: Gomma: a component-based infrastructure for managing and analyzing life science ontologies and their evolution. Journal of Biomedical Semantics 2, 6 (2011)
7. Rahm, E.: Towards large-scale schema and ontology matching. Schema matching and mapping pp. 3–27 (2011)

Hertuda Results for OEAI 2012

Sven Hertling

Technische Universität Darmstadt
hertling@ke.tu-darmstadt.de

Abstract. Hertuda is a very simple element based matcher. It shows that tokenization and a string measure can also yield in good results. It is an improved version of the first version submitted to the OAEI 2011.5.

1 Presentation of the system

1.1 State, purpose, general statement

Hertuda is a first idea of an element based matcher with a string comparison. It generates only homogeneous matchings, that are compatible with OWL Lite/DL. This means that classes, data properties and object properties are handled separately. As a result there are three thresholds that can be set independently. One for class to class, object to object property and data to data property. A simple overall threshold sets all sub-thresholds to the same value.

Over all concepts a cross product is computed. If the confidence of a comparison is higher than the threshold for this type of matching, then it is added to the resulting alignment. For each concept all labels, comments and URI fragments are extracted. Then these terms form a set. To compare two concepts, respectively sets of terms, each element of the first set is compared with each element of the second set. The best value is the similarity measure for these concepts.

A preprocessing step for term comparison is to tokenize it. All camel case terms or terms with underscores or hyphens in it, are split into single tokens and converted to lower case. Therefore *writePaper*, *write-paper* and *write_paper* will all result in two tokens, namely $\{write\}$ and $\{paper\}$.

Afterwards a similarity matrix is computed with the Damerau–Levenshtein distance [1, 2]. The average of the best mappings are then returned as the similarity between two token sets. Figure 1 depicts schematically the algorithm of Hertuda.

1.2 Specific techniques used

The final matching system contains of the string matching approach and a filter for removing alignments that are not considered in the reference alignment. The system is depicted in figure 2.

The filter removes all alignments that are true, but are not in the reference alignment. The removed mappings are mostly from upper level ontologies like *dublin core* or *friend of a friend*.

```
void function hertuda() {
  for each type in {class, data property, object property}
    for each concept cOne in ontology one
      for each concept cTwo in ontology two
        if(compareConcepts(cOne, cTwo) > threshould(type)){
          add alignment between cOne and cTwo
        }
}

float compareConcepts(Concept cOne, Concept cTwo) {
  for each termOne in {label(cOne), comment(cOne), fragment(cOne)}
   for each termTwo in {label(cTwo), comment(cTwo), fragment(cTwo)}
     conceptsMatrix[termOne, termTwo] = compareTerms(termOne, termTwo)

  return maximumOf(conceptsMatrix)
}

float compareTerms(String tOne, String tTwo) {
  tokensOne = tokenize(tOne)
  tokensTwo = tokenize(tTwo)

  tokensOne = removeStopwords(tokensOne)
  tokensTwo = removeStopwords(tokensTwo)

  for each x in tokensOne
   for each y in tokensTwo
    similarityMatrix[x, y] = damerauLevenshtein(x, y)

  return bestAverageScore(similarityMatrix)
}
```

Fig. 1. Algorithm for Hertuda

1.3 Adaptations made for the evaluation

There are no specific adaptions made. The overall threshold for a normalised Damerau–Levenshtein distance is set to 0.88.

1.4 Link to the system and parameters file

The tool version submitted to OAEI 2012 can be downloaded from `http://www.ke.tu-darmstadt.de/resources/ontology-matching/hertuda`

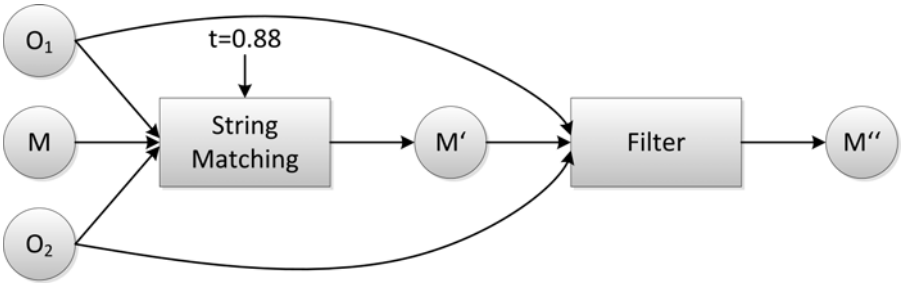


Fig. 2. Composition of matching algorithms of Hertuda. The string based approach and the filter are sequential composed.

2 Results

2.1 Benchmark

The implemented approach is only string based and works on the element level, whereby missing labels or comments or replaced terms by random strings has a high effect on the matching algorithm.

2.2 Anatomy

Hertuda has a higher recall than the *StringEquiv* from OEAI 2011.5 (0.673 to 0.622). Through the tokenization and also the string distance the precision is much lower (0.69 to 0.997). This yield in worse F-Measure for Hertuda (68.1 to 0.766).

2.3 Conference

The first version of Hertuda only compares the tokens for equality, whereas the new version computes a string similarity. Though the recall is a little bit higher than the first version, but the precision is lower. All in all, the F-Measure has increased by 0.01. This approach can find a mapping between *has_the_first_name* and *hasFirstName* with an similarity of 1.0.

2.4 Multifarm

Hertuda is not designed for multiligual matching. Nevertheless, some simple alignments are returned like $person(en) \equiv person(de)$.

2.5 Library

In the Library Track a relatively high recall has been achieved (0.925). Through splitting the words a very low precision value (0.465) was the result.

2.6 Large Biomedical Ontologies

Hertuda was only capable to match the small task for FMA-NCI and FMA-SNOMED. The large ones are not finished in time. The reason can be, that the complexity is to high trough the cross product of all concepts.

3 General comments

3.1 Comments on the results

The approach shows, that also simple string based algorithms can yield in good results. The improvement of version 1 is not much, but the recall was higher in many tracks. The precision was therefore lower, but it ends often in better F-Measure values.

3.2 Discussions on the way to improve the proposed system

To improve Hertuda it is possible to add more stop words in different languages. This helps by comparing two ontologies that have the same language, but this differs from English.

Another point is to set the threshold more precise and not one for all. It is also imaginable to set the threshold based on the matching ontologies. This will help to reduce the low precision in some tracks.

4 Conclusion

The results show that an string based algorithm can also produce good alignments. The recall of this version is in many cases much higher that the first version. Thus it is possible to use this matcher as a previous step of structural matchers.

References

1. Damerau, F.: A technique for computer detection and correction of spelling errors. Communications of the ACM **7**(3) (1964) 171–176
2. Levenshtein, V.: Binary codes capable of correcting deletions, insertions and reversals. In: Soviet Physics Doklady. Volume 10. (1966) 707

HotMatch Results for OEAI 2012

Thanh Tung Dang, Alexander Gabriel, Sven Hertling, Philipp Roskosch,
Marcel Wlotzka, Jan Ruben Zilke, Frederik Janssen, and Heiko Paulheim

Technische Universität Darmstadt
{janssen,paulheim}@ke.tu-darmstadt.de

Abstract. HotMatch is a multi-strategy matcher developed by a group of students at Technische Universität Darmstadt in the course of a hands-on training. It implements various matching strategies. The tool version submitted to OAEI 2012 combines different basic matching strategies, both element-based and structure-based, and a set of filters for removing faulty mappings.

1 Presentation of the system

1.1 State, purpose, general statement

HotMatch[1] has been developed by a group of students in the course of a semantic web hands-on training conducted at TU Darmstadt. The students were asked to develop and implement different matching algorithms. For OAEI 2012, we have combined a large number of those matching algorithms into one tool. To give an overview of our approaches, all matchers are depicted in figure 1. In contrast to *matchers*, *filters* are used to remove mapping elements found by previous matchers.

1.2 Specific techniques used

HotMatch provides a library of different matching algorithms and filters.

Matching Algorithms
ElementStringMatcher is a simple string-based, element-level matcher on the element level. All labels, URI fragments and comments are extracted and tokenized. As a second step some stopwords are removed. To get a similary measure of two concepts, a cross product of labels, fragments and comments is calculated with the Damerau–Levenshtein distance.

GraphbasedUseClassMatcher is a graph based matcher. It operates on the structural level and needs some input alignment to have an initial mapping between classes.

Figure 2 gives an example of the mapping candidates. The properties X and Y are matched if the domain and range are equals respectively are aligned with a previous matcher. The confidence of the new mapping between the two Properties is the mean value between the confidence of mapping A to C and B to D.

[1] For **H**ands-**o**n **t**raining matcher

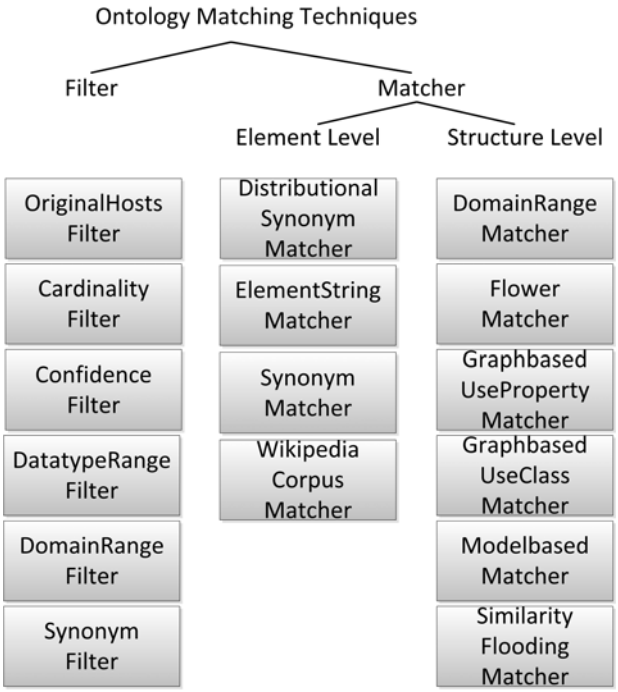


Fig. 1. Overview on the matching and filtering algorithms implemented in HotMatch.

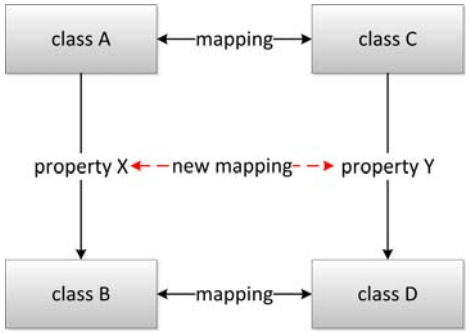


Fig. 2. New mapping of *GraphbasedUseClassMatcher*. Class A and C as well as B and D are already matched. Property X and Y is therefore also matched.

GraphbasedUsePropertyMatcher is a modification of *GraphbasedUseClassMatcher*. It uses properties from previous alignments instead of classes. If a property is matched from previous approaches, then the domain and range are also matched in a new alignment, inheriting the confidence mapping between the properties.

SimilarityFlooding implements the structural similarity flooding matching algorithm described in [3].

FlowerMatcher is a matching algorithm which combines a structural and an element-based approach. For each ontology class, its neighborhood (super and subclasses, properties that this class is a domain or range of) are regarded. From the names and labels of all the concepts in the neighborhood, a joint set of trigrams is computed. These sets are compared for determining the class similarity.

ModelbasedMatcher checks currently only if the union of the two ontologies plus the input mappings is valid. The implementation uses the *pellet* reasoner. In the future, this matcher is supposed to add extra mappings derived by reasoning, as well discard mappings that generate a contradiction.

DistributionSynonymMatcher and **WikipediaCorpusMatcher** are matchers using external resources, i.e., the online API *lanes*[2]. The distribution synonym matcher tries to identify synonyms based on distributional similarity, i.e., the similarity of the context in which two words occur [1]. The Wikipedia corpus matcher computes the percentage of Wikipedia pages on which two terms co-occur (similar to the approach discussed in [2]).

SynonymMatcher uses the online thesaurus *Big Huge Thesaurus*[3] to find mappings between concepts.

Filters

OriginalHostsFilter extracts the major host component of the input ontologies' URIs. If an alignment has other URI hosts than the major one, this alignment is removed. The remaining mappings are not changed. This filter is necessary, because an alignment like

$$\begin{gathered} < http://purl.org/dc/elements/1.1/description, \\ http://purl.org/dc/elements/1.1/description, \\ =, \\ 1.0 > \end{gathered}$$

is definitely true, but not contained in the reference alignments. In OAEI tracks, it will thus generate a false positive and reduce the mathcher's precision.

CardinalityFilter is a filter to enforce a $1:1$ alignment. If a resource from ontology one are matched to multiple resources from ontology two, then only the alignment with the highest confidence is selected. All other mappings are discarded. The same procedure is also applied for ontology two. The result of this filter is an alignment that relates each element from one ontology to at most one element from another ontology.

ConfidenceFilter is a simple filter that removes all alignments that have a smaller confidence than a given threshold.

[2] Language Analysis Essentials, `http://research.wilsonwong.me/lanes.html`

[3] `http://words.bighugelabs.com/`

DomainRangeFilter discards all alignments with non-matched domain and range. This is particularly useful for discarding inverses (e.g., *isReviewerOf* vs. *hasReviewer*), which receive high similarity scores with simple element-based techniques.

DatatypeRangeFilter checks only datatype properties. Matched properties hat have a different datatype (e.g., string vs. date) are discarded.

SynonymFilter has been implemented as a variant of the **SynonymMatcher** (see above). Since the latter has shown to produce a too large number of false positives (but with reasonable recall), it can also be used as a filter, e.g., on structural approaches for improving precision.

1.3 Adaptations made for the evaluation

The final matcher composition of the version submitted to OAEI 2012 is shown in figure 3. The threshold for confidence filter is set to $t = 0.7$. Note that not all matchers and filters discussed above are included in the final composition. We discarded all components that did not improve the system's accuracy and favored faster components over slower ones in case of ties.

All matchers are composed sequentially. The upper lane shows all matchers which generate new alignments. The lower one depicts all filters used to remove alignments that are not in the reference alignment to improve the precision value.

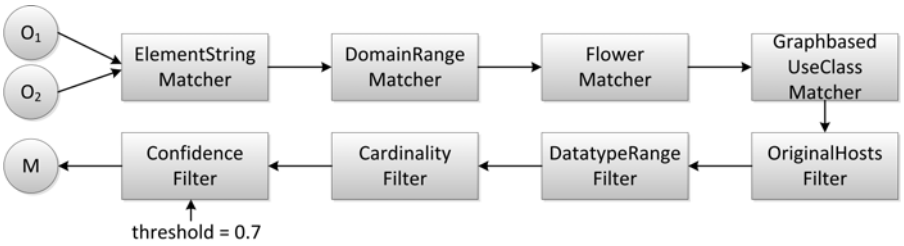


Fig. 3. Final composition for the evaluation

Although the filters only remove elements from the mapping generated by the matchers, they cannot be arbitrarily permuted. For example, the cardinality filter enforcing a 1:1 mapping will select the candidate with the highest threshold. If a mapping element with a higher threshold is filtered, e.g., by the OriginalHostsFilter, the selection will be different. Consider the following constellation for a mapping between ontology A and B, where B imports the FOAF ontology[4]:

$$< A\#person, B\#author, =, 0.7 > \quad (1)$$

$$< A\#person, foaf\#person, =, 0.8 > \quad (2)$$

[4] `http://xmlns.com/foaf/spec/`

Using the CardinalityFilter first would discard the first element, and the second one would be discarded by the OriginalHostsFilter. On the other hand, using the OriginalHostsFilter first would discard the second element, with the first one passing the CardinalityFilter.

1.4 Link to the system and parameters file

The tool version submitted to OAEI 2012 can be downloaded from `http://www.ke.tu-darmstadt.de/resources/ontology-matching/hotmatch`.

2 Results

2.1 Benchmark

HotMatch relies on string similarity to a large extent; although some structural measures are used later in the pipeline. Thus, it only performs well on those benchmark cases where names and labels are preserved. In particular, they show that the filters work quite effectively, since the precision only rarely drops below 0.95.

2.2 Anatomy

On the anatomy track, the performance of HotMatch is more or less the same as the string equivalence baseline[5]. In other words, the structure-based approaches do not improve the results much. This is not surprising as the structure-based approaches in HotMatch largely rely on `domain` and `range` definitions, which are not present in the Anatomy track. The reported runtime of 672 seconds shows an average behavior.

2.3 Conference

This track gives some insights into the strengths and weaknesses of *HotMatch*. In contrast to the anatomy track, the structure-based measures in HotMatch are capable of exploiting the `domain` and `range` definitions in the conference ontologies. For example, the structure-based algorithms provide some useful mappings, such as `hasAuthor = isWrittenBy` or `hasBeenAssigned = isReviewing`, but are also prone to produce false positives such as `Reviewer = MemberPC`, since both share a common super class. In terms of F-Measure, the results are comparable to *Baseline2*[6] (i.e., string matching with some pre-processing), but with a tendency to prefer recall over precision in comparison to that baseline, as the examples above show.

[5] `http://oaei.ontologymatching.org/2011.5/results/anatomy/index.html`

[6] `http://oaei.ontologymatching.org/2011.5/results/conference/index.html`

2.4 Multifarm

This matcher is not designed to work with multilingual ontologies. The results are accordingly low. Only some labels are equals in their translation like *person* in German as well as in English. Such resources are matched through string equality. Despite those occasional mappings, there is no correlation of the result quality and involved the languages' similarity – strangely enough, the best results are achieved for German-Chinese, two languages that are not known to be particularly similar.

2.5 Library

The mapping quality achieved by HotMatch on the library track is not as positive as on the other tracks. Possible reasons may be the absence of `domain` and `range` definitions (in fact, of properties in general), as for anatomy, and the presence of multi-lingual labels. As HotMatch does not respect languages, this may lead to false positives.

2.6 Large Biomedical Ontologies

HotMatch has been reported to have some problems of finishing the larger datasets in this track on time. As the matching process itself is rather light-weight, this may hint at efficiency issues of the implementation of HotMatch.

3 General comments

3.1 Comments on the results

The results show that with a multi-strategy approach using different simple matching strategies, reasonable results can be produced. There is a gap to more sophisticated systems – which is expected – but the results on the conference track also show that some of the more complex systems can be beaten.

3.2 Discussions on the way to improve the proposed system

One key feature of HotMatch is the ability to combine multiple matchers and filters. The final configuration submitted to OAEI has been found using extensive manual testing, however, it is a compromise which is supposed to produce reasonable results on most of the tracks.

Being able to individually assembling a configuration for each pair of ontologies would be an interesting option, thus, the system would clearly benefit from leveraging work in these fields [4, 5].

3.3 Comments on the OAEI 2012 Measures

In the current OAEI test cases, mapping elements that are correct but refer to concepts of other ontologies (like the example in Sect. 1.2) cause false positives, since they are not part of the reference alignment. In the HotMatch version for OAEI, we filter them manually, however, a real-world ontology matching system that returns those elements as well could equally make sense.

To circumvent this problem, the organizers might consider filtering mapping elements refering to concepts from other ontologies before computing precision.

4 Conclusion

In this paper, we have discussed the results for the HotMatch system, a multi-strategy matching system developed by students at Technische Universität Darmstadt in the course of a hands-on training. We have shown that the system provides reasonable results on most of the OAEI tracks and can compete with many state-of-the-art matching tools.

References

1. Harris, Z.S.: MathematicalStructuresOfLanguage. Wiley (1968)
2. Hertling, S., Paulheim, H.: Wikimatch - using wikipedia for ontology matching. In: Seventh International Workshop on Ontology Matching (OM 2012). (2012)
3. Melnik, S., Garcia-Molina, H., Rahm, E.: Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In: 18th International Conference on Data Engineering, IEEE (2002) 117–128
4. Mochol, M., Jentzsch, A., Euzenat, J.: Applying an analytic method for matching approach selection. In: Proceedings of the 1st International Workshop on Ontology Matching (OM-2006). (2006)
5. Ritze, D., Paulheim, H.: Towards an automatic parameterization of ontology matching tools based on example mappings. In: Sixth International Workshop on Ontology Matching. (2011)

LogMap and LogMapLt Results for OAEI 2012

Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, and Ian Horrocks

Department of Computer Science, University of Oxford
{ernesto,berg,ian.horrocks}@cs.ox.ac.uk

Abstract. We present the results obtained by our ontology matching system LogMap and its 'lightweight" variant called LogMapLt within the OAEI 2012 campaign. The LogMap project started in January 2011 with the objective of developing a scalable and logic-based ontology matching system. This is our third participation in the OAEI and the experience has so far been very positive.

1 Presentation of the system

LogMap [10, 14] is a highly scalable ontology matching system with built-in reasoning and inconsistency repair capabilities. LogMap also supports (real-time) user interaction during the matching process, which is essential for use cases requiring very accurate mappings. To the best of our knowledge, LogMap is the only matching system that (1) can efficiently match semantically rich ontologies containing tens (and even hundreds) of thousands of classes, (2) incorporates sophisticated reasoning and repair techniques to minimise the number of logical inconsistencies, and (3) provides support for user intervention during the matching process.

LogMap is also available as a "lightweight" variant called LogMapLt, which essentially skips all reasoning, repair and semantic indexation steps. Due to its simplicity, scalability and reasonable quality of its output, LogMapLt has been adopted as baseline in some OAEI tracks [19].

1.1 Technical challenges

Building a scalable, logic-based and interactive ontology matching presents important technical challenges. Moreover, these requirements are in some respects conflicting, and design choices require compromises between them. We next provide an overview of the technical challenges we have faced in the design of LogMap.

I. Computing Candidate Mappings. Computing mappings requires pairwise comparison of the entities in the vocabularies of the relevant ontologies (e.g., using a string matcher). This leads to a search space that is quadratic in the size of the ontologies (e.g., there are over 4 billion candidate mappings between FMA and NCI). For large ontologies, performing such huge number of pairwise comparisons is unfeasible in practice, even if we rely on the fastest available string matchers. Hence, reducing the search space of candidate mappings is a key challenge for a scalable ontology matching system.

II. Detection of unsatisfiable classes. Ontology $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$ resulting from the integration of $\mathcal{O}_1$ and $\mathcal{O}_2$ via mappings $\mathcal{M}$ may entail axioms that do not follow from

$\mathcal{O}_1$, $\mathcal{O}_2$, or $\mathcal{M}$ alone. Many such entailments correspond to unsatisfiable classes, which are due to either erroneous mappings or to inherent disagreements between $\mathcal{O}_1$ and $\mathcal{O}_2$. For example, the union of FMA, SNOMED and the UMLS [3] mappings between them (which are the result of careful manual curation) has over $6,000$ unsatisfiable classes [13], and the number of unsatisfiable classes may be even higher when mappings are not subject to manual curation. Although state-of-the-art OWL 2 reasoners can efficiently classify existing large-scale biomedical ontologies individually (e.g., ELK [16] can classify SNOMED in a few seconds and HermiT [21] can classify FMA in less than a minute), the integration of these ontologies via mappings leads to challenging classification problems [9] (e.g., no reasoner known to us can classify the integration of SNOMED and NCI via mappings).

III. Repair of unsatisfiable classes. Standard justification-based repair techniques (e.g., [15, 23, 8]) can be used to repair the identified unsatisfiable classes in $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$. These techniques have been implemented in mapping repair systems such as ContentMap [12] and Alcomo[1] [18]. The scalability problem, however, is exacerbated by the number of unsatisfiable classes to be repaired. For example, computing all justifications for just one out of the $6,000$ unsatisfiable classes in the integration of FMA-SNOMED via UMLS mappings requires, on average, over 9 minutes using HermiT — even with the optimisation proposed in [24]; doing this for all unsatisfiable classes would require more than 6 weeks.

IV. Expert feedback during the matching process is important for use cases requiring very accurate mappings; however, smooth interaction with domain experts imposes very strict scalability requirements. Furthermore, feedback requests to a human expert should not be overwhelming and should be used only when strictly needed. Hence, it is crucial to reduce the number of feedback requests, on the one hand, as well as the delay between successive requests, on the other hand.

1.2 Technical approach

In order to meet these challenges, we have relied on the following key elements in the design of LogMap (see [10, 14] for details).

Lexical indexation. An inverted index is used to store the lexical information contained in the input ontologies. This index is the key to addressing *challenge I* since it allows for the efficient computation of an initial set of mappings of manageable size. Similar indexes have been successfully used in information retrieval and search engine technologies [2].

Logic-based module extraction. The practical feasibility of unsatisfiability detection and repair critically depends on the size of the input ontologies. To reduce the size of the problem, we exploit ontology modularisation techniques. Ontology modules with well-understood semantic properties can be efficiently computed and are typically much smaller than the input ontology [5, 17].

[1] Note that Alcomo also implements incomplete reasoning and repair techniques.

Propositional Horn reasoning. The relevant modules in the input ontologies together with (a subset of) the candidate mappings are encoded in LogMap using a Horn propositional representation. LogMap implements the classic Dowling-Gallier algorithm for propositional Horn satisfiability [6, 7], which can be exploited to detect unsatisfiable classes in linear time. Such encoding, although incomplete, allows LogMap to address *challenge II* soundly and efficiently.

Axiom tracking and greedy repair. LogMap extends Dowling-Gallier's algorithm to track all mappings that may be involved in the unsatisfiability of a class. This extension is key to implementing a highly scalable greedy repair algorithm that can meet *challenge III*.

Semantic indexation. The Horn propositional representation of the ontology modules and the mappings are efficiently indexed using an interval labelling schema [1] — an optimised data structure for storing directed acyclic graphs (DAGs) that significantly reduces the cost of answering taxonomic queries [4, 22]. In particular, this semantic index allows us to answer many entailment queries over the input ontologies and the mappings computed thus far as an index lookup operation, and hence without the need for reasoning. The semantic index complements the use of a propositional encoding to address *challenges II-III* and it is the key to meeting *challenge IV*.

1.3 Adaptations made for the evaluation

LogMap's algorithm described in [10, 14] has been extended with basic functionalities to support matching of instance data.

LogMap's instance matching module is based on the same lexical indexation techniques used in LogMap to match classes. In order to discover additional instance mappings, LogMap also exploits the property assertions of the input ontologies to analise the structure of their ABoxes.

In order to minimise the number of logical errors caused by the instance mappings, LogMap's repair module is also used to detect and repair conflicts.

1.4 Link to the system and parameters file

LogMap[2] is open-source and released under GNU Lesser General Public License 3.0.[3] Latest components and source code are available from the LogMap's Google code page: `http://code.google.com/p/logmap-matcher/`.

LogMap distributions can be easily customized through a configuration file containing the matching parameters.

LogMap can also be used directly through an AJAX-based Web interface where matching tasks can be easily requested: `http://csu6325.cs.ox.ac.uk/`

[2] `http://www.cs.ox.ac.uk/isg/projects/LogMap/`
[3] `http://www.gnu.org/licenses/`

Table 1: Results for Benchmark track.

System	biblio			bench1			bench2			bench2			finance		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
LogMap	0.73	0.45	0.56	1.00	0.47	0.64	0.95	0.49	0.65	0.99	0.46	0.63	0.95	0.47	0.63
LogMapLt	0.79	0.50	0.59	0.95	0.50	0.66	0.95	0.50	0.65	0.95	0.50	0.65	0.90	0.52	0.66

Table 2: Results for Anatomy track.

System	P	R	F	Time (s)
LogMap	0.92	0.845	0.881	20
LogMapLt	0.963	0.728	0.829	6

2 Results

In this section, we present the results obtained by LogMap and LogMapLt in the OAEI 2012 campaign.

2.1 Benchmark track

Ontologies in this track have been synthetically generated. The goal of this track is to evaluate the matching systems in scenarios where the input ontologies lack important information (e.g., classes contain no meaningful URIs or labels).

Table 1 summarises the average results obtained by LogMap and LogMapLt. Note that the computation of candidate mappings in LogMap and LogMapLt heavily relies on the similarities between the vocabularies of the input ontologies; hence, there is a direct negative impact in the cases where the labels are replaced by random strings.

2.2 Anatomy track

This track involves the matching of the Adult Mouse Anatomy ontology (2,744 classes) and a fragment of the NCI ontology describing human anatomy (3,304 classes). The reference alignment has been manually curated, and it contains a significant number of non-trivial mappings.

Table 2 summarises the results obtained by LogMap and LogMapLt. The evaluation was run on a machine with 4GB RAM and 2 cores.

2.3 Conference track

The Conference track uses a collection of 16 ontologies from the domain of academic conferences [25]. These ontologies have been created manually by different people and are of very small size (between 14 and 140 entities). The track uses two reference alignments RA1 and RA2. RA1 contains manually curated mappings between a subset of the 120 ontology pairs evaluated in the track. RA2 contains composed mappings, based on the alignments in RA1, between all the ontology pairs.

Table 3 summarises the average results obtained by LogMap and LogMapLt. The last column represents the total runtime on generating all 120 alignments. Tests were run on a laptop with Intel Core i5 2.67GHz and 4GB RAM.

Table 3: Results for Conference track.

System	RA1 reference			RA2 reference			Time (s)
	P	R	F	P	R	F	
LogMap	0.82	0.58	0.68	0.77	0.53	0.63	211
LogMapLt	0.73	0.5	0.59	0.68	0.45	0.54	44

Table 4: Results for Library track.

System	P	R	F	Time (s)
LogMap	0.688	0.644	0.665	95
LogMapLt	0.577	0.776	0.662	21

2.4 Multifarm track

This track is based on the translation of the OntoFarm collection of ontologies into 9 different languages [20]. Both LogMap and LogMapLt, as expected, obtained poor results since they do not implement specific multilingual techniques.

2.5 Library track

The library track involves the matching of the STW thesaurus (6,575 classes) and the TheSoz thesaurus (8,376 classes). Both of these thesauri provide vocabulary for economic and social sciences. Table 4 summarises the results obtained by LogMap and LogMapLt. The track was run on a machine with 7GB RAM and 2 cores.

2.6 Large BioMed track

This track aims at finding alignments between large and semantically rich biomedical ontologies such as FMA, SNOMED, and NCI [11]. UMLS Metathesaurus has been selected as the basis for the track reference alignments [3]. Since the UMLS mappings together with the input ontologies lead to numerous unsatisfiable classes, two refinements of the UMLS mappings have also been considered as reference alignments. These refinements have been generated using LogMap's repair facility [10] and the Alcomo debugging system [18]. The track has been split into nine tasks involving different fragments of FMA, SNOMED, and NCI.

LogMap has been evaluated with two configurations in this track. LogMap's default algorithm computes an estimation of the overlapping between the input ontologies before the matching process, while the variant $\text{LogMap}_{\text{noe}}$ has this feature deactivated.

Tables 5-7 summarises the results obtained by LogMap, $\text{LogMap}_{\text{noe}}$ and LogMapLt. Precision and recall represent average values for the three reference alignments. The number of unsatisfiable classes as a consequence of reasoning (using HermiT [21]) with the input ontologies and the output mappings is also given.[4] Note that LogMap, unlike $\text{LogMap}_{\text{noe}}$, failed to detect and repair a few unsatisfiable classes in the SNOMED-NCI matching problem since they were outside the computed ontology fragments. The track was run on a server with 16 CPUs and allocating 15GB RAM.

[4] Since no OWL 2 reasoner can classify the integration of SNOMED and NCI via mappings [9], the Dowling-Gallier algorithm [6] for propositional Horn satisfiability was used instead.

Table 5: Results for the Large BioMed track: FMA-NCI tasks

Task 1: Small FMA and NCI fragments						
System	**Size**	**Unsat.**	**P**	**R**	**F**	**Time (s)**
LogMap	2,740	2	0.932	0.876	0.903	18
LogMap$_{noe}$	2,740	2	0.932	0.876	0.903	18
LogMapLt	2,483	2,104	0.945	0.806	0.870	8
Task 2: Big FMA and NCI fragments						
System	**Size**	**Unsat.**	**P**	**R**	**F**	**Time (s)**
LogMap	2,656	5	0.870	0.793	0.830	77
LogMap$_{noe}$	2,663	5	0.872	0.798	0.833	74
LogMapLt	3,219	12,682	0.729	0.806	0.766	29
Task 3: whole FMA and NCI ontologies						
System	**Size**	**Unsat.**	**P**	**R**	**F**	**Time (s)**
LogMap	2,652	9	0.860	0.783	0.819	131
LogMap$_{noe}$	2,646	9	0.866	0.787	0.825	206
LogMapLt	3,466	26,429	0.677	0.806	0.736	55

Table 6: Results for the Large BioMed track: FMA-SNOMED tasks

Task 4: Small FMA and SNOMED fragments						
System	**Size**	**Unsat.**	**P**	**R**	**F**	**Time (s)**
LogMap	6,164	2	0.910	0.667	0.769	65
LogMap$_{noe}$	6,363	0	0.910	0.688	0.784	63
LogMapLt	1,645	773	0.938	0.183	0.307	14
Task 5: Big FMA and SNOMED fragments						
System	**Size**	**Unsat.**	**P**	**R**	**F**	**Time (s)**
LogMap	6,292	0	0.833	0.623	0.712	484
LogMap$_{noe}$	6,450	0	0.837	0.642	0.727	521
LogMapLt	1,819	2994	0.848	0.183	0.302	96
Task 6: whole FMA and SNOMED ontologies						
System	**Size**	**Unsat.**	**P**	**R**	**F**	**Time (s)**
LogMap	6,312	10	0.828	0.621	0.710	612
LogMap$_{noe}$	6,406	10	0.816	0.621	0.706	791
LogMapLt	1,823	4938	0.846	0.183	0.301	171

Table 7: Results for the Large BioMed track: SNOMED-NCI tasks

Task 7: Small SNOMED and NCI fragments						
System	**Size**	**Unsat.**	**P**	**R**	**F**	**Time (s)**
LogMap	13,454	0*	0.897	0.649	0.753	221
LogMap$_{noe}$	13,525	0*	0.895	0.652	0.754	211
LogMapLt	10,947	61,269*	0.945	0.557	0.701	54
Task 8: Big SNOMED and NCI fragments						
System	**Size**	**Unsat.**	**P**	**R**	**F**	**Time (s)**
LogMap	12,142	3*	0.874	0.571	0.691	514
LogMap$_{noe}$	13,184	0*	0.879	0.624	0.730	575
LogMapLt	12,741	131,073*	0.812	0.557	0.661	104
Task 9: whole SNOMED and NCI ontologies						
System	**Size**	**Unsat.**	**P**	**R**	**F**	**Time (s)**
LogMap	13,011	16*	0.814	0.570	0.671	955
LogMap$_{noe}$	13,058	0*	0.811	0.570	0.670	1,505
LogMapLt	14,043	305,648*	0.737	0.557	0.634	178

Table 8: Results for Instance matching track.

System	Sandbox			IIMB		
	P	R	F	P	R	F
LogMap	0.94	0.94	0.94	0.94	0.91	0.93
LogMapLt	0.95	0.89	0.92	0.84	0.82	0.83

2.7 Instance matching

LogMap and LogMapLt have participated in the *Sandbox* and *IIMB* matching tasks. The SandBox and IIMB datasets have been automatically generated by introducing a set of controlled transformations in an initial ABox, as a result Sandbox and IIMB contains 11 and 80 synthetic ABoxes, respectively.

Table 8 summarises the average results obtained by LogMap and LogMapLt. The results are quite promising considering that this is the first participation of LogMap in this track. Nevertheless, there is still room for improvement in order to deal with more challenging tasks.

3 General comments and conclusions

Comments on the results. LogMap's main weakness is that the computation of candidate mappings relies on the similarities between the vocabularies of the input ontologies; hence, there is a direct negative impact in the cases where the ontologies are lexically disparate or do not provide enough lexical information.

Discussions on the way to improve the proposed system. LogMap is now a stable and mature system that has been made available to the community. There are, however, many exciting possibilities for future work. For example we aim at implementing multilingual features in order to be competitive in the Multifarm track. We also intend to extend LogMap's instance matching module with more sophisticated techniques.

Comments on the OAEI 2012 measures. Although the *mapping coherence* is a measure already used in the OAEI we consider that is not given the required weight in the evaluation. Thus, developers focus on creating matching systems that maximize the F-measure but they disregard the impact of the generated ouput in terms of logical errors.

Acknowledgements. This work was supported by the Royal Society, the EPSRC project LogMap and the EU FP7 projects SEALS and Optique. We also thank the organisers of the OAEI evaluation campaigns for providing test data and infrastructure and Anton Morant and Yujiao Zhou who have also contributed to the LogMap project in the past.

References

1. Agrawal, R., Borgida, A., Jagadish, H.V.: Efficient management of transitive relationships in large data and knowledge bases. In: SIGMOD Rec. 18. pp. 253–262 (1989)

2. Baeza-Yates, R.A., Ribeiro-Neto, B.A.: Modern Information Retrieval. ACM Press / Addison-Wesley (1999)
3. Bodenreider, O.: The unified medical language system (UMLS): integrating biomedical terminology. Nucleic Acids Research 32, 267–270 (2004)
4. Christophides, V., Plexousakis, D., Scholl, M., Tourtounis, S.: On labeling schemes for the Semantic Web. In: Int'l World Wide Web (WWW) Conf. pp. 544–555 (2003)
5. Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U.: Modular reuse of ontologies: Theory and practice. J. Artif. Intell. Res. 31, 273–318 (2008)
6. Dowling, W.F., Gallier, J.H.: Linear-time algorithms for testing the satisfiability of propositional Horn formulae. J. Log. Prog. 1(3), 267–284 (1984)
7. Gallo, G., Urbani, G.: Algorithms for testing the satisfiability of propositional formulae. J. Log. Prog. 7(1), 45–61 (1989)
8. Horridge, M., Parsia, B., Sattler, U.: Laconic and precise justifications in OWL. In: Int'l Sem. Web Conf. (ISWC). pp. 323–338 (2008)
9. Jiménez-Ruiz, E., Cuenca Grau, B., Horrocks, I.: On the feasibility of using OWL 2 DL reasoners for ontology matching problems. In: OWL Reasoner Evaluation Workshop (2012)
10. Jimenez-Ruiz, E., Cuenca Grau, B.: LogMap: Logic-based and Scalable Ontology Matching. In: Int'l Sem. Web Conf. (ISWC). pp. 273–288 (2011)
11. Jiménez-Ruiz, E., Cuenca Grau, B., Horrocks, I.: Exploiting the UMLS Metathesaurus in the Ontology Alignment Evaluation Initiative. In: E-LKR Workshop (2012)
12. Jiménez-Ruiz, E., Cuenca Grau, B., Horrocks, I., Berlanga, R.: Ontology integration using mappings: Towards getting the right logical consequences. In: Eur. Sem. Web Conf. (ESWC). pp. 173–187 (2009)
13. Jiménez-Ruiz, E., Cuenca Grau, B., Horrocks, I., Berlanga, R.: Logic-based assessment of the compatibility of UMLS ontology sources. J. Biomed. Sem. 2 (2011)
14. Jiménez-Ruiz, E., Cuenca Grau, B., Zhou, Y., Horrocks, I.: Large-scale interactive ontology matching: Algorithms and implementation. In: Eur. Conf. on Artif. Intell. (ECAI) (2012)
15. Kalyanpur, A., Parsia, B., Horridge, M., Sirin, E.: Finding all justifications of OWL DL entailments. In: Int'l Sem. Web Conf. (ISWC). pp. 267–280 (2007)
16. Kazakov, Y., Krötzsch, M., Simancik, F.: Concurrent classification of EL ontologies. In: Int'l Sem. Web Conf. (ISWC). pp. 305–320 (2011)
17. Konev, B., Lutz, C., Walther, D., Wolter, F.: Semantic modularity and module extraction in description logics. In: European Conf. on Artif. Intell. (ECAI). pp. 55–59 (2008)
18. Meilicke, C.: Alignment Incoherence in Ontology Matching. Ph.D. thesis, University of Mannheim (2011)
19. Meilicke, C., Svab-Zamazal, O., Trojahn, C., Jimenez-Ruiz, E., Aguirre, J., Stuckenschmidt, H., Cuenca Grau, B.: Evaluating ontology matching systems on large, multilingual and real-world test cases. In: ArXiv e-prints (2012), http://arxiv.org/abs/1208.3148v1
20. Meilicke, C., Castro, R.G., Freitas, F., van Hage, W.R., Montiel-Ponsoda, E., de Azevedo, R.R., Stuckenschmidt, H., Šváb-Zamazal, O., Svátek, V., Tamilin, A., Trojahn, C., Wang, S.: MultiFarm: a benchmark for multilingual ontology matching. J. Web Sem. (2012)
21. Motik, B., Shearer, R., Horrocks, I.: Hypertableau reasoning for description logics. J. Artif. Intell. Res. 36, 165–228 (2009)
22. Nebot, V., Berlanga, R.: Efficient retrieval of ontology fragments using an interval labeling scheme. Inf. Sci. 179(24), 4151–4173 (2009)
23. Schlobach, S., Huang, Z., Cornet, R., van Harmelen, F.: Debugging incoherent terminologies. J. Autom. Reasoning 39(3) (2007)
24. Suntisrivaraporn, B., Qi, G., Ji, Q., Haase, P.: A modularization-based approach to finding all justifications for OWL DL entailments. In: Asian Sem. Web Conf. (ASWC) (2008)
25. Šváb, O., Svátek, V., Berka, P., Rak, D., Tomášek, P.: OntoFarm: towards an experimental collection of parallel ontologies. In: Int'l Sem. Web Conf. (ISWC). Poster Session (2005)

MaasMatch results for OAEI 2012

Frederik C. Schadd, Nico Roos

Maastricht University, The Netherlands
{frederik.schadd, roos}@maastrichtuniversity.nl

Abstract. This paper summarizes the results of the participation of MaasMatch in the Ontology Alignment Evaluation Initiative (OAEI) of 2012. We provide a brief description of the techniques that have been applied, with the emphasis being on the utilized similarity measures and the performed improvements over the system that participated in the year 2011. Additionally, the results of the 2012 OAEI campaign will be discussed.

1 Presentation of the system

1.1 State, purpose, general statement

Sharing and reusing knowledge is an important aspect in modern information systems. Since multiple decades, researchers have been investigating methods that facilitate knowledge sharing in the corporate domain, allowing for instance the integration of external data into a company's own knowledge system. Ontologies are at the center of this research, allowing the explicit definition of a knowledge domain. With the steady development of ontology languages, such as the current OWL language [5], knowledge domains can be modelled with an increasing amount of detail.

The initial research of the MaasMatch framework focused on resolving terminological heterogeneities between ontology concepts, which is reflected in its initial selection of similarity measures. Recent research focused on further developing these techniques, while increasing its spectrum of similarity measures such that the system can be applicable in a wider area of matching tasks. The supported matching domain of ontologies for MaasMatch are limited to semi-large, meaning up to $\sim$2000 concepts per ontology, mono-lingual OWL ontologies, thus yielding predictable results for the Library and Multifarm tracks.

1.2 Specific techniques used

Various similarity measures covering differing categories have been applied in the current system. This subsection provides a brief explanation of each measure and how these are combined to extract the final alignment.

Syntactic Similarity MaasMatch currently utilizes a token-based measure for the purpose of determining the syntactic similarity between concepts. More specifically, concept names and labels are compared by computing the 3-grams [10] of their names and determining their similarity using the Jaccard [3] measure.

Structural Similarity As structural similarity a Name-Path similarity is used. Given a concept c, such a similarity collects the name of c and all ancestors of c, which is subsequently used as a basis for comparison. Given the nature of these strings, a hybrid similarity has been selected for this purpose. A hybrid similarity is defined as any similarity that relies on another similarity measure for its computation. Cohen et al. [1] researched a token-based framework for a hybrid distance. Given two strings s and t, the set of tokens $a_1, a_2, \ldots, a_K$ into which string s can be divided into and the set of tokens $b_1, b_2, \ldots, b_L$ into which string t can be divided into, a hybrid distance can be computed as follows:

$$sim(s,t) = \frac{1}{K} \sum_{i=1}^{K} \max_{j=1}^{L} sim'(a_i, b_j) \tag{1}$$

The hybrid similarity in MaasMatch utilizes the Levenshtein [4] similarity, to which a substring-based extension is applied. This extension functions similarly to the Winkler [11] extension, however is not limited to the size or location of the substring. This setup has been shown to outperform other variations of measures on the conference dataset and a record matching dataset [2]. Given two strings s and t, the longest common substring of s and t defined as $LCS(s,t)$ and a scaling factor S, sim' of our hybrid distance is computed as follows:

$$sim'(s,t) = Levenshtein(s,t) + \frac{LCS(s,t)}{\min(s,t)} \cdot S \cdot (1 - Levenshtein(s,t)) \tag{2}$$

Virtual Document Similarity A new similarity that is deployed in MaasMatch is the comparison of virtual documents representing ontology concepts, which are created by gathering the information contained within a concept and the information of its related neighbours according to a specific model. This approach has been pioneered by Qu et al. [7]. In essence, this approach uses a weighted combination of descriptions of concepts. A description of a concept is a weighted document vector describing the terms that occur in the concept description. The model of creating such a description allows for certain types of terms, such as the concept name, label or comments, to be weighted differently according to their perceived importance. Descriptions of related concepts are added to the description of a particular concept by multiplying the term weights of the related descriptions with a diminishing factor before merging the vectors. For a full description of this process, we recommend the reader to consult the works of Qu et al. [7].

Lexical Similarity This similarity has seen improvements, compared to its counterpart of the 2011 competition, with regard to its computing time. The similarity uses WordNet as a basic lexical resource, however utilizes virtual document similarities between ontology concepts and WordNet synsets in order to only assign synsets to concepts which accurately describe the meaning of that concept. Given two ontologies O_1 and O_2 that are to be matched, O_1 contains the sets of entities $E_x^1 = \{e_1^1, e_2^1, ..., e_m^1\}$, where x distinguishes between the set of classes, properties or instances, O_2 contains the sets

of entities $E_x^2 = \{e_1^2, e_2^2, ..., e_n^2\}$, and $C(e)$ denotes a collection of synsets representing entity e, the essential steps of our approach, performed separately can be described as follows:

1. For every entity e in E_x^i, compute its corresponding set $C(e)$ by performing the following procedure:
   (a) Assemble the set $C(e)$ with synsets that might denote the meaning of entity e.
   (b) Create a virtual document of e, and a virtual document for every synset in $C(e)$.
   (c) Calculate the document similarities between the virtual document denoting e and the different virtual documents originating from $C(e)$.
   (d) Discard all synsets from $C(e)$ that resulted in a low similarity score with the virtual document of e, using some selection procedure.
2. Compute the WordNet similarity for all combinations of $e^1 \in E_x^1$ and $e^2 \in E_x^2$ using the processed collections $C(e^1)$ and $C(e^2)$.

Figure 1 illustrates steps 1.b - 2 of our approach for two arbitrary ontology entities e^1 and e^2:

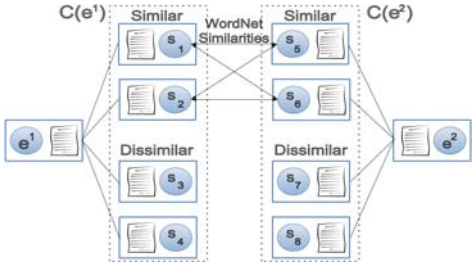


Fig. 1. Visualization of step 1.b-2 of the proposed approach for any entity e^1 from ontology O_1 and any entity e^2 from ontology 2.

Further details of the particular steps of this approach are illustrated in the works by Schadd et al. [9].

Aggregation and Extraction In our system, similarity matrices are aggregated by computing the average similarity measure of each pairwise combination of concepts, based on the computed similarity cube. The Naive descending extraction algorithm [6] is applied on the aggregated similarity matrix in order to determine the final alignment. At this point a confidence threshold can be applied in order to avoid producing alignments which do not satisfy a determined degree of confidence.

1.3 Adaptations made for the evaluation

While for practical applications it is recommended to apply a confidence boundary in the extraction step, this has been omitted for the evaluation system in order to provide

the possibility for the experimenters to conduct a more thorough analysis of the produced alignments, even if these have a low confidence value and would not be included in the final alignment under normal circumstances.

1.4 Link to the system and parameters file

MaasMatch and its corresponding parameter file is available on the SEALS platform and can be downloaded at http://www.seals-project.eu/tool-services/browse-tools.

2 Results

This section presents the evaluation of the OAEI2012 results achieved by MaasMatch. Evaluations utilizing ontologies exceeding the supported complexity range, such as the Library track, will be excluded from the discussion for the sake of brevity. Note that the evaluations of some of the tracks do not determine the optimal confidence threshold of the produced alignments such that correspondences with low confidence values are incorporated into the evaluations as well, resulting in lower performance measures compared to a normal execution environment.

2.1 Benchmark

The benchmark data set consists of several base ontologies which are matched with automatically altered versions of themselves. This makes it possible to establish under what condition a matcher performs well or poorly. Previous competitions used only a single ontology as base, with the alterations being done by hand. The current data set consists of several base ontologies such that a more varied spectrum of knowledge domains is utilized. The results of MaasMatch on the benchmark data set can be seen in Table 1.

Test Set	Precision	Recall	F-Measure
biblio	0.54	0.57	0.56
2	0.6	0.6	0.6
3	0.53	0.53	0.53
4	0.54	0.54	0.54
finance	0.59	0.6	0.59

Table 1. Aggregated harmonic means of the benchmark test sets.

From Table 1 it is observable that the results set a stark contrast in comparison to the competition of 2011 [8]. The continued development of our system was successful in increasing the recall of the produced alignments, however this came at a cost of reduced recall, yielding a similar f-measure when compared to the previous year. However, this evaluation does not take into account the confidence values provided with the

alignments, resulting in alignments with low confidence value being included in the evaluation. In a realistic scenario a pruning mechanism, for instance a simple cutoff rate, would be applied such that matches with low confidence values would not be included. As reported by the experimenter, pruning the alignments results in f-measure gains between 0.07 to 0.15, mostly due to a significant gain in precision, thus yielding significantly improved results over the MaasMatch system of 2011.

2.2 Anatomy

The anatomy data set consists of two large real-world ontologies from the biomedical domain, with one ontology describing the anatomy of a mouse and the other being the NCI Thesaurus, which describes the human anatomy. The results of this data set can be seen in Table 2.

Test Set	Precision	Recall	F-Measure
mouse-human	0.434	0.784	0.559

Table 2. Results of the anatomy data set.

Also the results of the anatomy data set have seen some drastic changes compared to the results of the previous year. The recall has been significantly improved, albeit at the cost of a significant proportion of precision. Overall, the f-measure has been improved by 0.11 over the results of the previous year [8].

2.3 Conference

The confidence data set consists of numerous real-world ontologies describing the domain of organizing scientific conferences. The results of this track can be seen in Table 3.

Test Set	Precision	Recall	F1-Measure
ra1	0.63	0.57	0.60
ra2	0.60	0.50	0.56

Table 3. Results of the conference data set.

For this data set, MaasMatch produced alignments of fairly balanced quality. The comparison to the standard reference alignments resulted in an f-measure of 0.6, which is a significant improvement compared to the same evaluation of the previous year. The evaluation using reference alignments which have been pruned using a consistency reason resulting in the recall being more affected than the precision of the alignments.

2.4 Large Biomedical Ontologies

This data set consists of several large scale ontologies, containing up to tens of thousands of concepts. While ontologies of such scale are not in the target domain of MaasMatch, due to the high computation complexity, some evaluation could still be performed, visible in Table 4.

Test Set	Precision	Recall	F-Measure
FMA-NCI Original UMLS	0.622	0.765	0.686
FMA-NCI Clean UMLS (LogMap)	0.606	0.778	0.681
FMA-NCI Clean UMLS (Alcomo)	0.597	0.788	0.679

Table 4. Results of the Large Biomedical Ontologies data set.

Among the varying evaluation methods, MaasMatch produced fairly consistent alignments when matching the FMA and NCI ontologies, all resulting in f-measures of approximately 0.68. Unfortunately, the remaining ontologies of this data set are outside of the supported complexity range, such that an alignment could not be computed within the given time frame. However, the results of the completed tasks indicate that our system is already capable of producing alignments of high quality in this domain, thus improving its efficiency, for instance by applying partitioning techniques, should result in an overall satisfying performance during the next evaluation.

2.5 Multifarm

The Multifarm data set is based on ontologies from the OntoFarm data set, that have been translated into a set of different languages in order to test the multi lingual capabilities of a specific system. Currently, the similarities employed by MaasMatch are not suitable in a multi-lingual matching problem, thus yielding predictably poor results.

Test Set	Precision	Recall	F-Measure
type I	0.02	0.14	0.03
type II	0.14	0.14	0.14

Table 5. Aggregated results of the Multifarm data set.

In Table 5, aggregation measures are separated into heterogeneous ontologies translated into different languages (type I) and homogeneous ontologies translated into different languages (type II). While the recall is unchanged for both matching types, the precision if positively influenced for homogeneous matching tasks.

3 General comments

3.1 Comments on the results

Overall, our system has seen improvements across various tracks, aided by the incorporation of additional similarity measures as well as the further development of the already existing measures. While the results of the previous year were high in precision and low in recall, the results of this year's participation demonstrate a more balanced measure of precision and recall, with both measures usually having a similar value.

3.2 Discussions on the way to improve the proposed system

The first area of improvement would consist of expanding the supported domain of matching problems, such that large scale or multi-lingual ontologies can be matched as well. Matching large scale ontologies would require the development of partitioning techniques in order to reduce the computational complexity of a matching task, preferably without impacting the results.

3.3 Comments on the SEALS platform

While the SEALS platform is a convenient tool for competition purposes, it would be nice to see its capabilities expanded such that evaluations can be automatically performed for research purposes, such that for instance any matching tool that is uploaded is automatically evaluated on the different available data sets.

3.4 Comments on the OAEI 2011 procedure

This years competition has seen some confusion whether or not the participants should omit post processing measures, such as cutoff based alignment pruning, given that some tracks perform automatic thresholding in order to generate the best possible alignments. However, the reported results of the benchmark data set did not include automatic thresholding, thus yielding the impression that the systems performs worse than it actually does. It would be preferable to have a clear statement on this matter and that each track is being evaluation according to the same policy.

3.5 Comments on the OAEI 2011 measures

An important part of the scientific method is the ability of recreating experimental results. Some tracks aggregate precision, recall and f-measure using the harmonic mean. However, given that the ranges of these 3 values lie in the interval of $[0, 1]$, it is possible that values of 0 would be incorporated in the evaluation, which in turn would yield a division by 0 due the reciprocal being computed of these values. It is currently unclear how this is circumvented and how exactly the measures are aggregated, making it very difficult to replicate experiments outside the OAEI environment. Thus it would be preferable to incorporate a detailed explanation on the computation and especially aggregation of the computed measures, even if this means including the same text in each year's proceedings.

4 Conclusion

This paper describes the 2012 participation of MaasMatch in the OAEI campaign, in which considerable improvements have been observed in the benchmark, anatomy and conference tracks, which have been evaluated in the previous year. New tracks were introduced with matching problems outside of the currently supported matching domain, however we intend to expand the capabilities of our system such the new types of problems can be tackled as well.

References

1. W. W. Cohen, P. Ravikumar, and S. E. Fienberg. A comparison of string distance metrics for name-matching tasks. In *Proc. IJCAI-03 Workshop on Information Integration on the Web (IIWeb-03)*, pages 73–78, 2003.
2. M. Hermans and F. C. Schadd. A generalization of the winkler extension and its application for ontology mapping. In *Proceedings Of The 24th Benelux Conference on Artificial Intelligence (BNAIC 2012)*, 2012.
3. P. Jaccard. Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin del la Société Vaudoise des Sciences Naturelles*, 37:547–579, 1901.
4. V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. Technical Report 8, 1966.
5. D. L. McGuinness and F. van Harmelen. OWL web ontology language overview. W3C recommendation, W3C, February 2004.
6. C. Meilicke and H. Stuckenschmidt. Analyzing mapping extraction approaches. *The Second International Workshop on Ontology Matching*, 2007.
7. Y. Qu, W. Hu, and G. Cheng. Constructing virtual documents for ontology matching. In *Proceedings of the 15th international conference on World Wide Web*, WWW '06, pages 23–31, New York, NY, USA, 2006. ACM.
8. F. C. Schadd and N. Roos. Maasmatch results for oaei 2011. In *Proc. 6th ISWC workshop on Ontology Matching (OM)*, pages 171–178, 2011.
9. F. C. Schadd and N. Roos. Coupling of wordnet entries for ontology mapping using virtual documents. In *Proceedings of the ISWC'12 International Workshop OM-2012*, 2012. Accepted Paper.
10. C. E. Shannon. A mathematical theory of communication. *SIGMOBILE Mob. Comput. Commun. Rev.*, 5(1):3–55, January 2001.
11. W. E. Winkler. String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage. Technical report, 1990.

MEDLEY Results for OAEI 2012

Walid Hassen

University of Tunis El Manar - Faculty of Sciences of Tunis
Computer Science Department - LIPAH
Campus Universitaire, 1060 Tunis, Tunisia
walidhassenlipah@gmail.com

Abstract. MEDLEY is an alignment method based on lexical and structural treatments. This method includes a specific technique to deal with multilingual ontologies. This paper introduces MEDLEY and summarizes the results for OAEI 2012.

1 Presentation of the system

MEDLEY can be presented as an OWL ontology alignment method that relies on simple similarity metrics. Each ontology pair, can be transformed into graphs structures. This means that links are OWL primitives and nodes are classes, properties, and individuals. The algorithm includes a lexical, structural treatment. Each node can be matched with few ones, then MEDLEY select pairs that maximize the global similarity value.

1.1 State, purpose, general statement

MEDLEY generates alignments between OWL-DL ontologies based on simple lexical metrics and structures matching between links of each node (class, property, instance). Specific treatment is applied for multilinguality issue, using a dictionnary to find equivalence between concepts labelled in diffrent natural languages.

1.2 Specific techniques used

Each entity in the first ontology is aligned each entity in the second, in a primary step, in lexical metrics, then in structural treatment. The algorithm reiterate this process for all ontologies's concepts.

- Lexical treatment : q-gram [1] and levenshtein [2] measures were used to calculate the similarity measures between nodes. In addition, treatments and tokenization stemmatisation were conducted.
- Structural treatment :If an entity belongs to a given ontology has a neighbor that is already part of the alignment set, then the node that neighbor is aligned to must be a neighbor of any prospective match for this entity.

1.3 Adaptations made for the evaluation

The MEDLEY method deals with three test suites used in the Ontology Alignment Evaluation Initiative (OAEI 2012). The method was wrapped in a certain folder structure to be evaluated locally after being integrated in the SEALS platform. The package contains all the libs files required by the method and a zipped .jar file that acts as a bridge.

1.4 Link to the system and parameters file

The release of the MEDLEY method and the parameter file used for OAEI 2012 are located at https://github.com/medley.

2 Results

In this section, we present the results obtained by MEDLEY in the OAEI 2012.

2.1 Benchmark

The benchmark tests sets can be divided into eight groups: 101, 20x, 22x, 23x, 24x, 25x, 26x and 30x. For each group the mean values of precision and recall are computed. Table 1 shows the values of the evaluation metrics. Tables 1, 2 and 3 recapitulate the obtained values for this track.

Table 1. Results on Biblio

Test group	Precision	Recall	F-Measure
101	0.72	1.0	0.84
20x	0.43	0.4	0.408
22x	0.716	1.0	.988
23x	0.781	1.0	0.853
24x	0.633	0.572	0.571
25x	0.51	0.4	0.421
26x	0.322	0.357	0.31

2.2 Conference

In scenario 1, MEDLEY have 0.54 of precision and 0.50, with 0.52 as recall an f-measure about 0.52. In scenario 2, MEDLEY performs 0.59 of precision, 0.42 recall and 0.49 of f-measure.

Table 2. Results on Benchmark 2

Test group	Precision	Recall	F-Measure
101	1.00	1.00	1.00
20x	0.697	0.4	0.493
22x	0.998	1.0	1.0
23x	0.995	1.0	1.0
24x	0.787	0.57	0.63
25x	0.757	0.439	035
26x	0.611	0.354	0.435

Table 3. Results on Benchmark 3

Test group	Precision	Recall	F-Measure
101	0.79	1.00	0.88
20x	0.568	0.4	0.454
22x	0.805	1.0	0.888
23x	0.88	1.0	0.93
24x	0.715	0.571	0.609
25x	0.642	0.4	0.463
26x	0.695	0.352	0.398

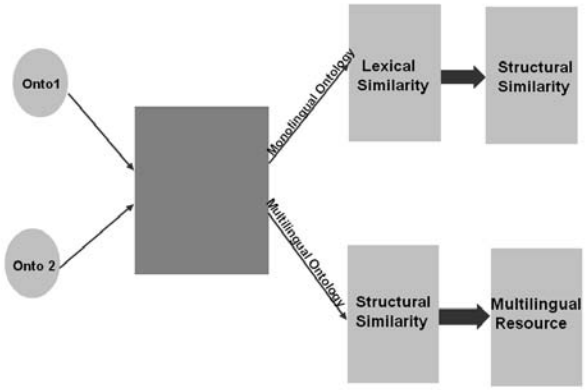


Fig. 1. MEDLEY components

2.3 Multifarm

For treating multilingual ontologies, our method uses an external resource as sketched by figure 1 for the translation stage[1]. Tables 4, 5, 6, 7, 8, 9 and 10 summarize the results.

[1] http://www.freelang.com/dictionnaire/index.php

Table 4. Group **(cz)** as source ontology

Test group	Precision	Recall	F-Measure
cz-de	0.51	0.07	0.13
cz-en	0.33	0.09	0.14
cz-es	0.43	0.07	0.12
cz-fr	0.33	0.05	0.09
cz-nl	0.33	0.06	0.10
cz-pt	0.46	0.08	0.13
cz-ru	0.00	0.00	NaN

Table 5. Group **(de)** as source ontology

Test group	Precision	Recall	F-Measure
de-en	0.40	0.10	0.15
de-es	0.43	0.09	0.15
de-fr	0.40	0.09	0.14
de-nl	0.38	0.09	0.15
de-pt	0.43	0.09	0.15
de-ru	0.00	0.00	NaN

Table 6. Group **(en)** as source ontology

Test group	Precision	Recall	F-Measure
en-es	0.54	0.48	0.51
en-fr	0.62	0.61	0.61
en-nl	0.56	0.42	0.48
en-pt	0.57	0.51	0.54
en-ru	0.05	0.00	0.00

Table 7. Group **(es)** as source ontology

Test group	Precision	Recall	F-Measure
es-fr	0.31	0.04	0.08
es-nl	0.21	0.03	0.05
es-pt	0.50	0.11	0.18
es-ru	0.02	0.00	0.00

3 General comments

We participate this year for the first time in OAEI and see the result obtained by our method. The evaluation and comparison of ontology alignment and schema matching components as OAEI is very useful for the development of such

Table 8. Group **(fr)** as source ontology

Test group	Precision	Recall	F-Measure
fr-nl	0.45	0.10	0.16
fr-pt	0.35	0.08	0.14
fr-ru	0.00	0.00	NaN

Table 9. Group **(nl)** as source ontology

Test group	Precision	Recall	F-Measure
nl-pt	0.31	0.07	0.11
nl-ru	0.03	0.00	0.00

Table 10. Group **(pt)** as source ontology

Test group	Precision	Recall	F-Measure
pt-ru	0.03	0.00	0.00

3.1 Discussions on the way to improve the proposed system

MEDLEY is still a primary work that needs to be adressed on few levels, notably, to deal with greater ontologies.

4 Conclusion

In this paper, we presented MEDLEY as an alignment method. The new proposed method MEDLEY, shows a special focus on multilinguality. The alignment process is based on examining the structures and the informative wealth on each ontlogy pair to align.

References

1. Ukkonen, E.: Approximate string-matching with Q-GRAMS and maximal matchs. Theoretical Computer Science **92**(1) (1992) 191–211
2. Levenshtein, I.V.: Binary codes capables of corrections, deletions, insertions and reversals. Soviet Physics-Doklady **10**(8) (1966) 707–710

OMReasoner: Using Multi-matchers and Reasoner for Ontology Matching: results for OAEI 2012

Guohua Shen, Changbao Tian, Qiang Ge, Yiquan Zhu, Lili Liao, Zhiqiu Huang, Dazhou Kang

Nanjing University of Aeronautics and Astronautics, Nanjing, China
{ghshen,cbtian,qge,yqzhu,llliao,zqhuang,dzkang}@nuaa.edu.cn

Abstract. Ontology matching produces correspondences between entities of two ontologies. The **OMReasoner** is unique in that it creates an extensible framework for combination of multiple individual matchers, and reasons about ontology matching by using description logic reasoner. It handles ontology matching in semantic level and makes full use of the semantic part of OWL-DL instead of structure. This paper describes the result of **OMReasoner** in the OAEI 2012 competition in two tracks: benchmark and conference.

1 Presentation of the system

Ontology matching finds correspondences between semantically related entities of the ontologies. It plays a key role in many application domains.

Many approaches to ontology matching have been proposed: the implementation of match may use multiple match algorithms or matchers, and the following largely-orthogonal classification criteria are considered [1-3]: schema-level and instance-level, element-level and structure-level, syntactic and semantic, language-based and constraint-based.

Most approaches focus on syntactic aspects instead of semantic ones. OMReasoner achieves the matching by means of reasoning techniques. Still, this approach includes strategy of combination of (mainly syntactical) multi-matchers (e.g., EditDistance matcher, Prefix/Suffix matcher, WordNet matcher) before match reasoning.

1.1 State, purpose, general statement

The matching process can be viewed as a function *f*.

$$A' = f(O1, O2, A, p, r)$$

Where *O1* and *O2* are a pair of ontologies as input to match, *A* is the input alignment between these ontologies and *A*' is new alignment returned, *p* is a set of parameters (e.g., weight *w* and threshold τ) and *r* is a set of oracles and resources.

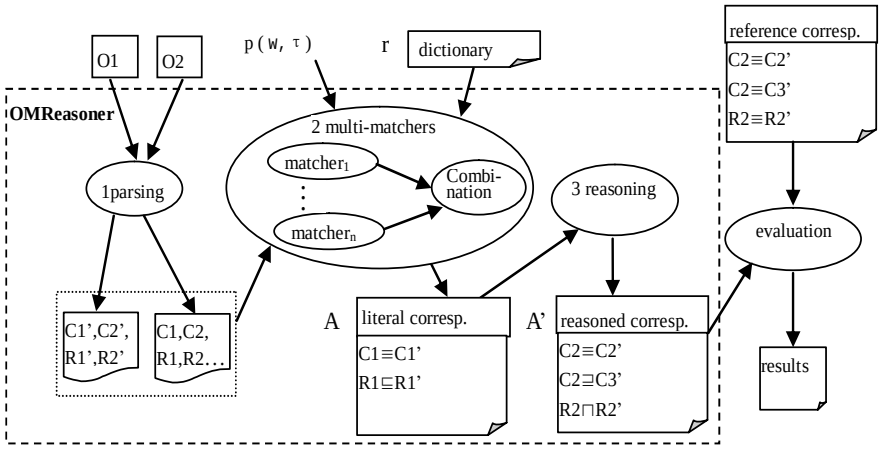


Fig.1. Ontology matching in OMReasoner

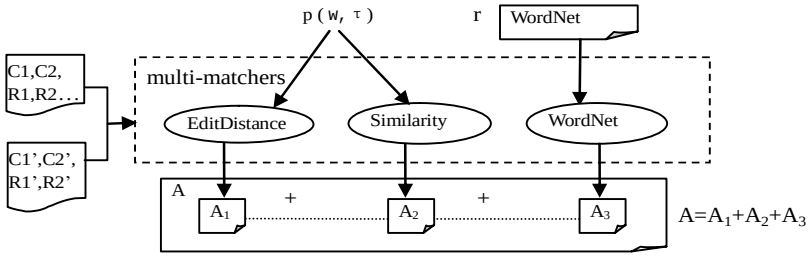


Fig.2. Instances of multi-matchers in OMReasoner

The OMReasoner achieved ontology alignment as following three steps (see Fig.1):

1. Parsing: we can achieve the classes and properties of ontologies by using ontology API: Jena.
2. Combination of multiple individual matchers: the literal correspondences (e.g. equivalence) can be produced by using multiple match algorithms or matchers, for example, string similarity measure (prefix, suffix, edit distance) by string-based, constrained-based techniques. Also, some semantic correspondences can be achieved by using some external dictionary: *WordNet*. Then the multiple match results can be combined by weighted summarizing method. The framework of multi-matchers combination is supported, which facilitates inclusion of new individual matchers.
3. Reasoning: the further semantic correspondences can be deduced by using DL reasoner, which uses literal correspondences produced in step 2 as input.

Finally, we evaluate the results against the reference alignments, and compute two measures: precision and recall.

In OMReasoner, the framework for multi-matchers is flexible, and any new individual matcher can be included. Now, the instances of multi-matchers include *EditDistance*, *Similarity* and *WordNet* (see Fig.2).

1.2 Specific techniques used

OMReasoner includes summarizing algorithm to combine the multiple match results. The combination can be summarized over the n weighted similarity methods (see formula 1), where w_k is the weight for a specific method, and $sim_k(e1,e2)$ is the similarity evaluation by the method.

$$sim(e1,e2) = \sum_{k-1}^{n} w_k sim_k(e1,e2) \quad (1)$$

OMReasoner uses semantic matching methods like *WordNet* matcher and description logic (DL) reasoning.

WordNet[1] is an electronic lexical database for English, where various senses (possible meanings of a word or expression) of words are put together into sets of synonyms. Relations between ontology entities can be computed in terms of bindings between WordNet senses. This individual matcher uses an external dictionary: WordNet to achieve semantic correspondences.

Another important matcher uses edit distance, which is a measure of the similarity between two words. Based on this value, we calculate the morphology analogous degree by using some math formula.

All the results of each individual matcher will be normalized before combination. OMReasoner employs DL reasoner provided by Jena. OMReasoner includes external rules to reason about the ontology matching.

2 Results：a comment for each dataset performed

There are 46 alignment tasks in benchmark data set and 21 alignment tasks in conference data set. We test the data sets with OMReasoner and present the results in Table 1, Table 2, Fig 3 and Fig 4. The average measures (precision, recall and F-Measure) of Benchmark are 0516, 0.379 and 0.419 respectively. The average measures of Conference are 0.159, 0.506 and 0.266 respectively. In conclusion, the precision, recall and F-Measure are not satisfying. However, we will improve it in the future.

2.1 Benchmark

We evaluated the results against reference alignments, and obtained precision varies from 0 to 0.949, and recall varies from 0 to 1.000, F-Measure varies from 0 to 0.990. Some measures are zero, because the reference alignments are a little bit strange. For example, *aqdsq* in dataset 248 is equivalent to some class in dataset 101.

[1] http://wordnet.princeton.edu/

Label	O1-O2	Prec.	Rec	f-Measure
B1	101-101	0.919	0.588	0.754
B2	101-103	0.919	0.588	0.754
B3	101-104	0.919	0.588	0.754
B4	101-202	0	0	0
B5	101-204	0	0	0
B6	101-204	0.917	0.567	0.739
B7	101-205	0.133	0.062	0.207
B8	101-206	0.540	0.278	0.527
B9	101-207	0.551	0.278	0.527
B10	101-208	0.917	0.567	0.739
B11	101-210	0.600	0.310	0.555
B12	101-221	0.919	0.588	0.754
B13	101-222	0.914	0.570	0.741
B14	101-223	0.919	0.588	0.754
B15	101-224	0.919	0.588	0.754
B16	101-225	0.919	0.588	0.754
B17	101-228	0.868	1.000	0.990
B18	101-230	0.949	0.514	0.690
B19	101-232	0.919	0.588	0.754
B20	101-233	0.868	1.000	0.990
B21	101-236	0.868	1.000	0.990
B22	101-237	0.914	0.570	0.741
B23	101-238	0.919	0.587	0.716
B24	101-239	0.853	1.00	0.9211
B25	101-240	0.868	1.00	0.929
B26	101-241	0.868	1.00	0.929
B27	101-246	0.794	0.931	0.857
B28	101-247	0.868	1.00	0.929
B29	101-248	0	0	0
B30	101-249	0	0	0
B31	101-250	0	0	0
B32	101-251	0	0	0
B33	101-252	0	0	0
B34	101-253	0	0	0
B35	101-254	0	0	0
B36	101-257	0	0	0
B37	101-258	0	0	0
B38	101-259	0	0	0
B39	101-260	0	0	0
B40	101-261	0	0	0
B41	101-262	0	0	0
B42	101-265	0	0	0
B43	101-266	0	0	0
B44	101-301	0.800	0.203	0.324
B45	101-302	0.833	0.3125	0.455
B46	101-304	0	0	0

Table.1. Match results in the Benchmark track

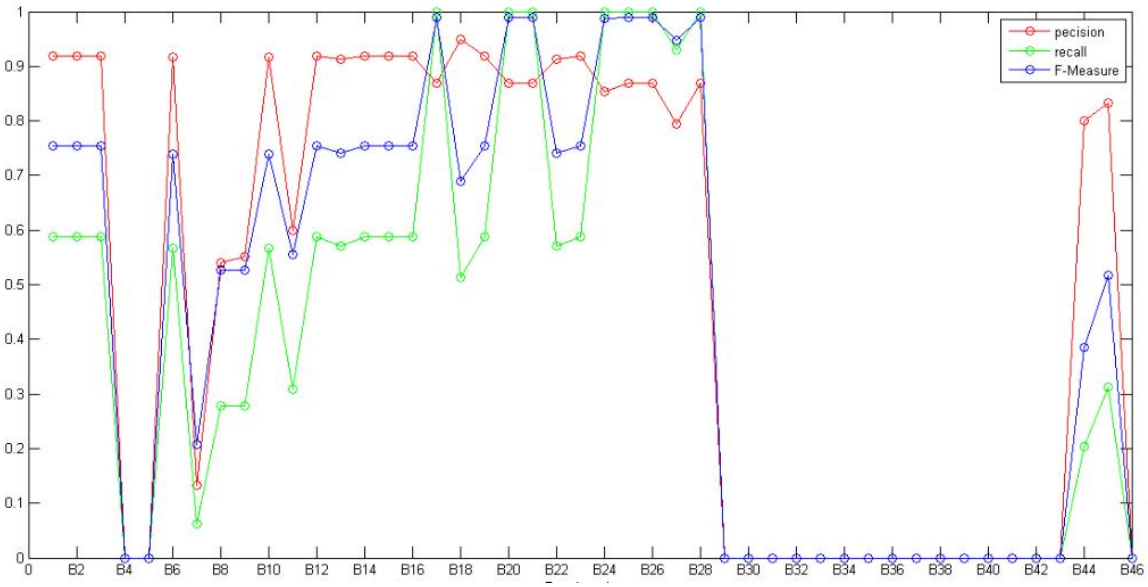


Fig.3. Comparison of match results in Benchmark

2.2 Conference

We evaluated the results against reference alignments, and obtained precision varies from 0.083 to 0.281, and recall varies from 0.296 to 1.000, F-Measure varies from 0.113 to 0.509.

Label	O1-O2	Prec.	Rec	F-Measure
C1	cmt-edas	0.190	0.615	0.360
C2	cmt-ekaw	0.146	0.545	0.282
C3	cmt-iasted	0.251	1.000	0.489
C4	cmt-sigkdd	0.281	0.750	0.509
C5	edas-ekaw	0.179	0.414	0.332
C6	edas-iasted	0.112	0.455	0.219
C7	edas-sigkdd	0.120	0.400	0.232
C8	ekaw-iasted	0.083	0.600	0.165
C9	ekaw-sigkdd	0.191	0.727	0.363
C10	iasted-sigkdd	0.172	0.667	0.331
C11	cmt-conference	0.149	0.412	0.219
C12	cmt-confOf	0.172	0.313	0.222
C13	conference-confOf	0.212	0.467	0.292
C14	conference-edas	0.111	0.368	0.171
C15	conference-ekaw	0.138	0.296	0.188
C16	conference-iasted	0.068	0.333	0.113
C17	conference-sigkdd	0.186	0.533	0.276

C18	confOf-edas	0.214	0.409	0.281
C19	confOf-ekaw	0.136	0.300	0.188
C20	confOf-iasted	0.095	0.444	0.157
C21	confOf-sigkdd	0.129	0.571	0.211

Table.2. Match results in the Conference track

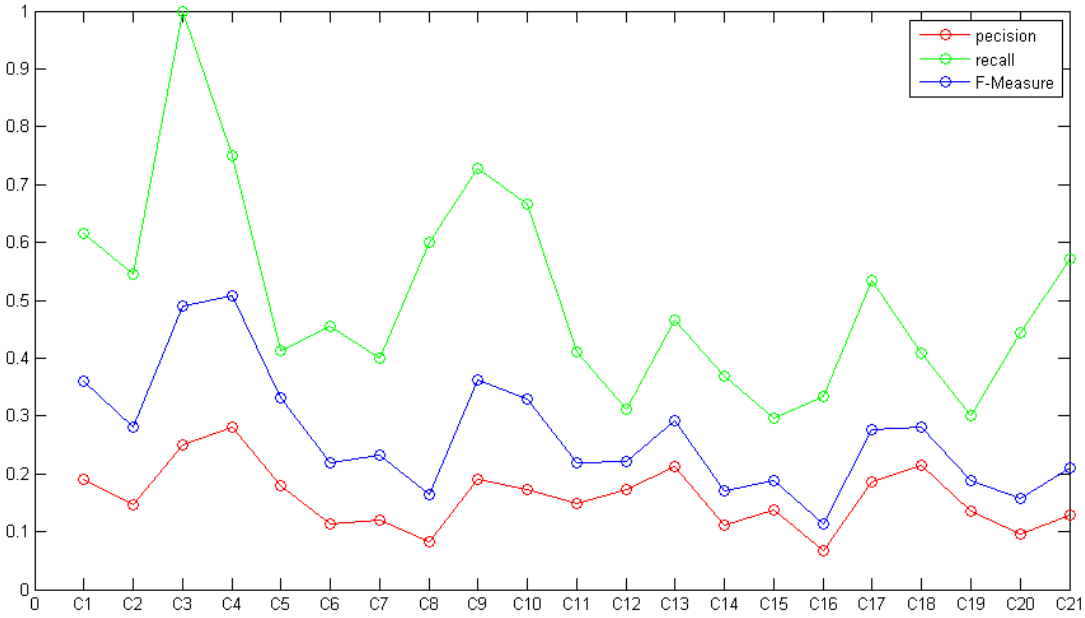


Fig.4. Comparison of match results in Conference

3 General comments

3.1 Comments on the results

The precision of results is not good enough, because only a few individual matchers are included.

The measures in Benchmark are better than those in Conference. The major reason is that the structure similarity of ontology is not considered in our tool.

3.2 Discussions on the way to improve the proposed system

The performance of inference relies on the literal correspondences heavily, so more accurate results which are exported from multi-matchers will greatly enhance the results of our tool.

Some probable approaches to improving our tool are listed as follow:

1. Adopt more flexible strategies in multi-matchers combination instead of just weighed sum.
2. Add some pre-processes, such as separating compound words, before words are imported into matchers.
3. Take comments and label information of ontology into account, especially when the name of concept is meaningless.
4. Improve the algorithm of some matchers.
5. More different matchers can be included.

Another problem in our tool is that we ignore structure information among ontology at the present stage. And we will improve it in the future.

3.3 Comments on the OAEI 2012 procedure

OAEI procedure arranged everything in good order, furthermore SEALS platform provides a uniform and convenient way to standardize and evaluate our tool.

4 Conclusion

In this paper, we presented the results of the OMReasoner system for aligning onltologies in the OAEI 2012 competition in two tracks: benchmark and conference. The combination strategy of multiple individual matchers and DL reasoner are included in our approach. This is the second time we participate the OAEI, the results is still not satisfying and we will improve it in the future.

References

1. Rahm, E. and Bernstein, P.: A survey of approaches to automatic schema matching. *The VLDB Journal*, ,10(4): 334--350(2001).
2. Shvaiko, P. and Euzenat, J.: A survey of schema-based matching approaches. *Journal on Data Semantics (JoDS)* IV, 146--171(2005).
3. Kalfoglou, Y. and Schorlemmer, M.: Ontology mapping: the state of the art. *The Knowledge Engineering Review Journal*, 18(1):1--31, (2003).
4. Shvaiko, P.: Iterative Schema-based Semantic Matching. PhD, University of Trento, (2006)
5. Jian, N., Hu, W., Cheng, G. et al: Falcon-AO: Aligning Ontologies with Falcon. *In: Proceedings of the K-CAP Workshop on Integrating Ontologies* (2005)

6. Do, H. and Rahm, E.: COMA- a system for flexible combination of schema matching approaches. *In: Proceedings of the International Conference on Very Large Databases*, 610--621.(2002)
7. Giunchiglia, F., Shvaiko, P., and Yatskevich, M.: S-Match: an algorithm and an implementation of semantic matching. *In: Proceedings of the European Semantic Web Symposium*, 61--75.(2004)
8. Kalfoglou, Y. and Schorlemmert, M.: If-map: an ontology mapping method based on information flow theory. *In: Proceedings of ISWC'03, Workshop on Semantic Integration*, (2003)
9. Bouquet, P., Serafini, L., and Zanobini, S.: Semantic coordination: A new approach and an application. *In: Proceedings of the International Semantic Web Conference*, 130--145.(2003)
10. Baader, F., Calvanese, D., McGuinness, D., et al.: The description logic handbook: theory, implementations and applications. *Cambridge University Press*, (2003)
11. Ehrig, M., Sure, Y.: Ontology mapping - an integrated approach. *In Proceedings of the European Semantic Web Symposium (ESWS)*, 76--91, (2004)
12. RacerPro User Guide. http://www.racer -systems. com, 2005
13. Do, H., Melnik, S., Rahm, E.: Comparison of Schema Matching Evaluations. *In: Proceedings of the 2nd Intl. Workshop on Web Databases*, Erfurt, Germany:,221--237(2002)
14. Shen, G., Jin, L., Zhao, Z., Jia, Z., He, W. and Huang, Z. : OMReasoner: using reasoner for ontology matching: results for OAEI 2011. *In Proceedings of* the 6th International Workshop on Ontology Matching.

Optima+ Results for OAEI 2012

Uthayasanker Thayasivam, Tejas Chaudhari, and Prashant Doshi

THINC Lab, Department of Computer Science, University of Georgia, Athens, Georgia 30602
{uthayasa,tejas,pdoshi}@cs.uga.edu

Abstract. In this report, we present the results of Optima+ in the Ontology Alignment Evaluation Initiative (OAEI) 2012. We mainly foucused on three tracks Benchmark, Conference, and Anatomy. However we were eavluated in all the tracks of the campaign offered in SEALS platform: Benchmark, Conference, Anatomy, Multifarm, Library, and LargeBioMed. We present the new and improved implementation of the Optima algorithm, Optima+ and its results for all the tracks offered within SEALS platform. Optima+ is the latest version of Optima , aimed to perform faster and better. Importantly, we match the highest f-measure (0.65) obtained for the conference track in last year's campaign. Moreover, this year we debut in large ontology tracks: Anatomy and Library aided by a naive divide and conquer approach.

1 Presentation of the system

The increasing popularity and utility of the semantic web increase the number of ontologies in the web. The applications such as web service compositions and semantic web search which utilize these ontologies demand a means to align these ontologies. At present we witness numerous ontology alignment algorithm and tools, that includes more than fifty ontology matching tools in SEALS platform [6] and many more which are not yet reported in SEALS platform [12, 2]. They can be broadly identified using their similarity measures, alignment algorithm and alignment extraction technique. We present a fully automatic general purpose ontology alignment tool called Optima+ , a new and improved implementation of its ancestor Optima [4].

Optima alignment process starts by generating a seed alignment using the lexical attributes of concepts (classes and properties) of the given ontology pair. Then it searches the space of candidate alignments in an iterative fashion and finds the best alignment which maximizes the likelihood. This likelihood estimation exploits the heuristic that the chance of a node pair in correspondence increases if their children are already mapped. Optima algorithm utilizes the lexical similarity between nodes within its structural matching such that its algorithm interlaces both structural and lexical attributes of nodes to arrive at an alignment. We brief out the formal model of an ontology as utilized by Optima and the alignment algorithm adopted by Optima in the next two subsections.

1.1 Ontology Model

The ontology alignment problem is to find a set of correspondences between two ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$. Because ontologies may be modeled as labeled graphs (though

with some possible loss of information), the problem is often cast as a matching problem between such graphs. An ontology graph, $\mathcal{O}$, is defined as, $\mathcal{O} = \langle V, E, L \rangle$, where V is the set of labeled vertices representing the entities, E is the set of edges representing the relations, which is a set of ordered 2-subsets of V, and L is a mapping from each edge to its label. Let M be the standard $|V_1| \times |V_2|$ matrix that represents the match between the two graphs $\mathcal{O}_\infty = \langle V_1, E_1, L_1 \rangle$, $\mathcal{O}_\in = \langle V_2, E_2, L_2 \rangle$:

$$M = \begin{bmatrix} m_{11} & m_{12} & \cdots & m_{1|V_2|} \\ m_{21} & m_{22} & \cdots & m_{2|V_2|} \\ . & . & \cdots & . \\ m_{|V_1|1} & m_{|V_1|2} & \cdots & m_{|V_1||V_2|} \end{bmatrix} \tag{1}$$

Each assignment variable in M is,

$$m_{a\alpha} = \begin{cases} 1 \text{ if } f(x_a) = y_\alpha : x_a \in V_1, y_\alpha \in V_2 \\ 0 \text{ otherwise} \end{cases}$$

Where $f(\cdot)$ represents the correspondence between the two ontology graphs. Consequently, M is a binary matrix representing the match.

1.2 EM-based Alignment Algorithm

Optima formulates the problem of inferring a match between two ontologies as a maximum likelihood problem, and solves it using the technique of expectation-maximization (EM) originally developed by Dempster et al. [3]. It implements the EM algorithm as a two-step process of computing expectation followed by maximization, which is iterated until convergence. The expectation step consists of evaluating the expected log likelihood of the candidate alignment given the previous iteration's alignment:

$$Q(M^i|M^{i-1}) = \sum_{a=1}^{|V_1|} \sum_{\alpha=1}^{|V_2|} Pr(y_\alpha|x_a, M^{i-1}) \times \ logPr(x_a|y_\alpha, M^i)\pi_\alpha^i \tag{2}$$

Where x_a and y_α are the entities of ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$, respectively, and π_α^i is the prior probability of y_α. $Pr(x_a|y_\alpha, M^i)$ is the probability that node x_a is in correspondence with node y_α given the match matrix M^i. The prior probability is computed using the following equation,

$$\pi_\alpha^i = \frac{1}{|V_1|} \sum_{a=1}^{|V_1|} Pr(y_\alpha|x_a, M^{i-1})$$

The generalized maximization step involves finding a match matrix, M_*^i, that improves on the previous one:

$$M_*^i = M^i \in \mathcal{M} : \ Q(M^i|M_*^{i-1}) \geq Q(M_*^{i-1}|M_*^{i-1}) \tag{3}$$

1.3 Optima+

Optima+ is a new and improved redesign of Optima to achieve a better alignment, yet in significantly less time. It adopts the block coordinate descent (BCD) technique for iterative ontology alignment proposed by us [14] to improve the convergence of the iterative process. Briefly, Optima+ is an optimized and efficient implementation of Optima algorithm. The new features Optima+ brings are 1) Block coordinate descent 2) Improved similarity calculation 3) Improved alignment extraction and 4) Large ontology matching. In the following four sub-sections we describe these four features.

Block Coordinate Descent For Optima Optima+ improve its performance by extending the Optima algorithm with the block coordinate descent (BCD) technique proposed in [14]. This technique helps to speed up its convergence. Let S denote a block of coordinates, which is indexed by a non-empty subset of $\{1, 2, \ldots, N\}$. We may define a set of such blocks as, $B = \{S_0, S_1, \ldots, S_C\}$, which is a set of subsets each representing a coordinate block with the constraint that, $S_1 \cup S_2 \cup \ldots \cup S_C = \{1, 2, \ldots, N\}$. Now, in each iteration, Optima+ (BCD enhanced Optima) chooses a block of the match matrix, $M^i_{S_c}$, and its expected log likelihood is estimated. It chooses the blocks in a sequential manner such that all the blocks are iterated in order. Equation 2 is modified to estimate the expected log likelihood of the block of a candidate alignment as:

$$Q_S(M^i_{S_c}|M^{i-1}) = \sum_{a=1}^{|V_{1,c}|} \sum_{\alpha=1}^{|V_2|} Pr(y_\alpha|x_a, M^{i-1}) \times \ logPr(x_a|y_\alpha, M^i_{S_c})\ \pi^i_{\alpha,c} \quad (4)$$

Here, $V_{1,c}$ denotes the set of entities of ontology, $\mathcal{O}_1$, participating in the correspondences included in S_c. Notice that the prior probability, $\pi^i_{\alpha,c}$, is modified as well to utilize just $V_{1,c}$ in its calculations.

The generalized maximization step now involves finding a match matrix block, $M^i_{S_c,*}$, that improves on the previous one:

$$M^i_{S_c,*} = M^i_{S_c} \in \mathcal{M}_{S_c} : Q_S(M^i_{S_c,*}|M^{i-1}_*) \geq Q_S(M^{i-1}_{S_c,*}|M^{i-1}_*) \quad (5)$$

Here, $M^{i-1}_{S_c,*}$ is a part of M^{i-1}_*. At iteration i, the best alignment matrix, M^i_*, is formed by combining the block matrix, $M^i_{S_c,*}$, which improves the Q_S function as defined in Eq. 5 with the remaining from the previous iteration, $M^{i-1}_{\bar{S}_c,*}$, unchanged.

An important heuristic, which has proven highly successful in ontology alignment, matches parent entities in two ontologies if their respective child entities were previously matched. This motivates grouping together those variables, $m_{a\alpha}$ in M, into a coordinate block such that the x_a participating in the correspondence belong to the same height leading to a partition of M. The height of an ontology node is the length of the shortest path from a leaf node. Let the partition of M into the coordinate blocks be $\{M_{S_0}, M_{S_1}, \ldots, M_{S_C}\}$, where C is the height of the ontology $\mathcal{O}_1$. Thus, each block is a submatrix with as many rows as the number of entities of $\mathcal{O}_1$ at a height and number of columns equal to the number of all entities in $\mathcal{O}_2$. For example, the correspondences between the leaf entities of $\mathcal{O}_1$ and all entities of $\mathcal{O}_2$ will form the block, M_{S_0}.

Similarity measures Similarity has become a classical tool for ontology matching. Similarity measure between ontological concepts such as classes and properties, is commonly a measure in the range of $[0, 1]$ represents how similar the two concepts are. The similarity measures used in the context of ontology matching can be broadly categorized into lexical similarity and structural similarity. Lexical similarity measures use the lexical properties of a concept (URIs, labels, names, and comments) to measure the similarity between the concepts while structural similarity measures exploit the graph matching algorithms to derive the similarity measure. The lexical similarity used in Optima+ between two concepts C_1 and C_2 is defined as,

$$Sim(C_1, C_2) = Max \left\{ \begin{array}{l} SimLex(Label\text{-}C_1, Label\text{-}C_2), \\ SimLex(Name\text{-}C_1, Name\text{-}C_2), \\ Cos(Comment\text{-}C_1, Comment\text{-}C_2) \end{array} \right\} \quad (6)$$

Where $Label$-C_1,$Name$-C_1, and $Comment$-C_1, are the label, name and comment of the concept C_1. As shown in Eq. 7 below the lexical similarity between the phrases P_1 and P_2 is,

$$SimLex(P_1, P_2) = Max \left\{ \begin{array}{l} LinSim(P_1, P_2), CosSim(P_1, P_2), \\ SWSim(P_1, P_2), NWSim(P_1, P_2), \\ LevSim(P1, P2) \end{array} \right\} \quad (7)$$

Here, LinSim is the popular similarity measure introduced by Lin [7] and CosSim is the gloss based cosine similarity described in [15]. These two similarity measures requires a lexical database like WordNet [9]. Optima+ uses WordNet version 3.0 for OAEI 2012 along with the information content database provided by [11]. SWSim is the Smith-Waterman [13] similarity measure and NWSim is the Needleman-Wunsch [10] similarity measure. LevSim is the similarity measure that is the inverse of Levenshtein distance between the phrases.

Alignment Extraction Alignment extraction is the process of pruning a set of correspondences in an alignment to achieve a minimal and consistent alignment. A minimal alignment is achieved by removing the correspondences which can be inferred by an existing correspondence. A consistent alignment is achieved by resolving conflicting correspondences. Optima+ adopts a simple heuristic based alignment extraction process, which is described below,

- For each class-correspondence (N_1, N_2) in the alignment, any correspondence among the children of N_1 and children of N_2 is removed.
- For each class-correspondence (N_1, N_2) in the alignment, any correspondence which maps children of N_1 to parent of N_2 or children of N_2 to parent of N_1 is removed if its similarity is less than the similarity of N_1 and N_2.
- If a concept is mapped to more than one concept then, we select the correspondence with highest similarity (Max_{Sim}) and remove all other correspondences which are less than a predefined threshold T_1. We also remove all other correspondences with similarity less than the $Max_{Sim} - \delta$. Here δ is a user configurable value in the range of $[0, 0.5]$.

Large Ontology Matching The time complexity of Optima to align Ontology O_1 of size $|O_1|$ and O_2 of size $|O_2|$ is $(|O_1| \times |O_2|)^2$ [4]. Hence, despite its efficient implementation in Optima+ , it still takes significantly longer time to match larger ontologies. We solve this problem using a naive divide and conquer approach. The large ontology matching is triggered if number of classes in one of the ontology exceeds a user configurable threshold (for this campaign it is set to 600 named classes). Optima+ partitions the ontology using a structural partitioning algorithm and matches every block from first ontology with every block from the second ontology separately. Finally, it merges all the block-alignments together as final alignment. The partitioning algorithm employed in Optima+ is based on breadth first tree traversal described in [4].

1.4 State, purpose, general statement

Optima+ is a general purpose ontology alignment tool capable of matching English language ontologies described in OWL, RDFS/RDF, and N3.

1.5 Specific techniques used

As described earlier, Optima+ employs a variety of similarity measures, a simple alignment extraction and large ontology matching using a naive divide and conquer approach.

1.6 Adaptations made for the evaluation

We made couple of changes to the alignment extraction process for this campaign. First, we filtered the correspondences between imported concepts even though they have been directly used within the ontologies. Second, we implemented the heuristics mentioned in the sub-section 1.3 to make the alignment minimal. The default alignment extraction of optima is not as strict as the one configured for this campaign.

1.7 Link to the system and parameters file

A detail presentation of the system, its configuration and parameters used for this campaign and results can be found at `http://thinc.cs.uga.edu/thinclabwiki/index.php/OAEI_2012`.

2 Results

Optima+ is evaluated in all the six tracks under SEALS platform in OAEI 2012 though, we only focused in benchmark, conference and anatomy tracks. For this report the results for all these tracks are summarized except for large biomedical track. Optima+ could not successfully finish aligning the large biomedical track due to a fatal error. Detailed results for individual tracks and test cases can be found at `http://thinc.cs.uga.edu/thinclabwiki/index.php/OAEI_2012`.

2.1 benchmark

The Benchmark test library consists of 5 different test suites [8]. Each of the test suits is based on individual ontologies, consists of number of test cases. Each test case discards a number of information from the ontology to evaluate the change in the behavior of the algorithm. There are six categories of such alterations – changing name of entities, suppression or translation of comments, changing hierarchy, suppressing instances, discarding properties with restrictions or suppressing all properties and expanding classes into several classes or vice versa. Suppressing entities and replacing their names with random strings results into scrambled labels of entities. Test cases from 248 till 266 consist of such entities with scrambled labels. Table. 1 shows Optima+ 's performance in benchmark track on, 100 series test cases, 200 series test cases without scrambled labels test cases and all the scrambled labels test cases. The average precision for Optima+ is 0.95 while average recall is 0.83 for all the test cases in 200 series except those with scrambled labels. For test cases with scrambled labels, the average recall is dropped by 0.53 while precision is dropped only by 0.04. When labels are scrambled, lexical similarity becomes ineffective. For Optima+ algorithm, structural similarity stems from lexical similarity hence scrambling the labels makes the alignment more challenging for Optima+ . Result is 46% decrease in average F-Measure from 0.85 to 0.46. This trend of reduction in precision, recall and f-measure can be observed throughout the benchmark track. For all the test suits, test cases with scrambled labels resulted into lower precision, recall and f-measure. Optima+ 's algorithm faces difficulties in aligning ontologies with low or no lexical similarity.

	Bibliography			2			3			4			Finance		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
100 Series	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
201-247	0.88	0.85	0.85	1	0.84	0.87	0.97	0.88	0.89	0.93	0.77	0.79	0.96	0.8	0.83
248-266	0.65	0.35	0.43	1	0.36	0.46	0.98	0.38	0.49	0.96	0.34	0.43	0.96	0.38	0.49

Table 1. Performance of Optima+ in OAEI 2012 for benchmark track

2.2 anatomy

Previous year, Optima could not sucessfully complete aliging anatomy track. This year, with the help of large ontology matching process, Optima+ is able to sucessfully align ontologies of this track. In anatomy track, Optima+ yields 0.854 precision and 0.584 recall in 6460 seconds. We hope with bio medical lexical databases like Unified Medical Language System (UMLS) [1] Optima+ could improve its recall.

2.3 conference

For this track, Optima+ achieves recall of 0.68 and precision of 0.62. Both the recall and the precision are improved compared to the performance of Optima in OAEI 2011. Overall there is 81% increase in F-Measure compared to OAEI 2011. This makes Optima+ , to tie the top performer in OAEI 2011[5] in terms of F-Meaure(0.65). Table 2 lists the harmonic means for precision, recall and f-measure along with total runtime for conference track of Optima in OAEI 2011 and Optima+ in OAEI 2012.

The performance improvement in conference track arises from the improved similarity measure and the alignment extraction (Section 1.3). Optima+ also utilizes improved design and optimization techniques to reduce the runtime. The runtimes reported in the Table 2 cannot be compared directly as the underlying systems used for evaluations differ. However, the runtime improvement from 15+ hours to around 23 minutes is perspicuous.

Year	Precision (H-mean)	Recall (H-mean)	F-Measure (H-mean)	Total Runtime
2011	0.26	0.60	0.36	15hrs
2012	0.62	0.68	0.65	1349sec

Table 2. Comparison between performances of Optima+ in OAEI 2012 and Optima in OAEI 2011 for conference track

2.4 multifarm

Since Optima+ focus only on English language ontologies, it gives low performance in this track as expected. However it is interesting to notice that Optima+ yields an average recall of 1.0 with an average precision of 0.01.

2.5 library

Library is another large ontology matching track in OAEI 2012. Optima+ attains a precision of 0.321 and a recall of 0.072 in 37,457 seconds.

3 General comments

Last year Optima debuted the OAEI campaign with promising results. However it took too long to finalize the alignment process. This year we redesigned the Optima algorithm to complete the alignment process faster and were able to speed it from minutes to seconds. Additionally, we implemented a naive divide and conquer approach to tackle the large ontology matching problem.

Optima+ matches the last year's best f-measure (0.65) in conference track, and gives 0.87 f-measure on average for benchmark track excluding the scrambled labeled test cases. However, as revealed in benchmark track Optima+ heavily relies on lexical features of ontologies to align them. In large ontology tracks (anatomy and library) Optima+ struggles to perform well as it performed in other tracks (conference and benchmark). We suppose that a dedicated alignment extraction is needed to merge the results of blocks in large ontology matching process.

We are aiming to improve our f-measure for large ontology matching by improving the entire large ontology matching process. Specifically, we would like to introduce an exclusive alignment extraction process for large ontology matching. Further, we want to find an optimum partition strategy for BCD technique which yields better alignment yet faster. On top of these, extending the current similarity measure calculation with more useful similarity measures and lexical databases would help Optima+ to improve its f-measure. Though there is an inherent means to align instances using Optima algorithm, Optima+ implementation is not yet fully capable of matching instances. In its next versions, we expect it to be able to match instances as well.

4 Conclusion

In this report we present the results of Optima+ in OAEI 2012 campaign in six tracks including Benchmark, Conference, Anatomy, Multifarm, Library, and LargeBioMed. We also present the new and redesigned implementation of Optima , Optima+ . Optima+ shows impressive performance in benchmark track, but struggles to align ontologies with scrambled labels. However, it matches the top f-measure of last year's conference track. It debuted in large ontology tracks (anatomy and library) with promising results. In future we want to participate in more tracks, especially instance matching tracks. More importantly, we wish to leverage our performance in large ontology tracks to attain a higher f-measure.

References

1. Bethesda. Umls reference manual. `http://www.ncbi.nlm.nih.gov/books/NBK9676/`, 2009.
2. S. Castano, A. Ferrara, and S. Montanelli. Matching ontologies in open networked systems: Techniques and applications. *Journal on Data Semantics (JoDS)*, V, 2005.
3. A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society.*, 39:1–38, 1977.
4. P. Doshi, R. Kolli, and C. Thomas. Inexact matching of ontology graphs using expectation-maximization. *Web Semantics*, 7(2):90–106, 2009.
5. Jérôme Euzenat, Alfio Ferrara, and et al. Results of the ontology alignment evaluation initiative 2011. In *Ontology Matching Workshop ISWC*, 2011.
6. Asuncion Góez-Pérez. Seals. `http://www.seals-project.eu/`, 2012.
7. D. Lin. An information-theoretic definition of similarity. In *ICML*, pages 296–304, 1998.
8. Jose Luis. Benchmark test library. `http://oaei.ontologymatching.org/2012/benchmarks/index.html`, 2012.
9. G. A. Miller. Wordnet: A lexical database for english. In *CACM*, pages 39–41, 1995.
10. Saul B. Needleman and Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443 – 453, 1970.
11. Ted Pedersen and Siddharth Patwardhan. Wordnet::similarity - measuring the relatedness of concepts. In *AAAI*, pages 1024–1025, 2004.
12. Quentin Reul and Jeff Z. Pan. Kosimap: Ontology alignments results for oaei 2009. In *Ontology Matching Workshop ISWC*, pages –1–1, 2009.
13. T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. In *JMB*, volume 147, pages 195–197, 1981.
14. Uthayasanker Thayasivam and Prashant Doshi. Improved convergence of iterative ontology alignment using block-coordinate descent. In *AAAI*, pages 150–156, 2012.
15. M. Yatskevich and F. Giunchiglia. Element level semantic matching using wordnet. Technical report, University of Trento, 2007.

SBUEI: Results for OAEI 2012

Aynaz Taheri, Mehrnoush Shamsfard

Computer Engineering Department, Shahid Beheshti University, Tehran, Iran
ay.taheri@mail.sbu.ac.ir,m-shams@sbu.ac.ir

Abstract. In this paper, we describe our system, SBUEI, for instances coreference resolution between various sources even with heterogeneous schemas. It is the first participation of SBUEI in instance matching track of Ontology Alignment Evaluation Initiative campaign. We present the results of SBUEI in the 2012 OAEI competition in two tracks: Sandbox and IIMB. SBUEI considers the instance coreference resolution in both schema and instance levels. The process of matching is applied to both levels consecutively to let the system discover identical instances.

1 Presentation of the system

Linked data resources have influential roles in conducting the future of semantic web. In recent years, different data providers have produced many data sources in Linking Open Data (LOD) cloud upon different schemas. Increases in the amount of linked data in LOD is not the only challenge of publishing linked data; rather, matching and linking the linked data resources are also equally important. The fourth rule of publishing linked data in [1] explains the necessity of linking URIs to each other. In the web of linked data, there are obviously many different kinds of schemas in various linked data resources. Therefore, we confront with schema heterogeneity in order to do coreference resolution. The importance of this issue motivated us to create a new system, SBUEI, for entity coreference resolution.

SBUEI deals with the both problems of instance matching and schema matching. SBUEI proposes an interleaving of instance and schema matching steps to find coreferences or unique identities in two sources. This approach is applicable to find unique identities in two linked data sources. SBUEI, unlike systems such as [2, 3, 4] -which uses just instance matching- or systems such as [5, 6] -which use just schema matching- exploits both levels of instance and schema matching. The main difference between SBUEI and other systems like [7], which exploit both levels, is that SBUEI exploits an interleaving of them while [7] exploits them sequentially one after the other (starts instance matching after completing schema matching). SBUEI utilizes schema matching results in instance matching and use the instance matching results in order to direct matching in schema level. SBUEI also has a new approach for instance matching.

1.1 State, purpose, general statement

SBUEI begins matching process by receiving two similar concepts of two ontologies called anchors. In fact, the inputs of SBUEI are two ontologies, two data sets of instances and the anchors (two equivalent concepts from the ontologies [8]).

Fig. 1 shows an example of performing SBUEI. In this figure two ontologies, *O1* and *O2* are represented. Each of them has a set of instances (*I1* and *I2*) . *a1* and *b1* are the anchors which are the two equivalent concepts of two ontologies. SBUEI begins the work with confidence to equality of *a1* and *b1* and starts searching instances of two concepts *a1* and *b1* to find instances with unique identity. This task is done by a new coreference resolution algorithm, described in [9].

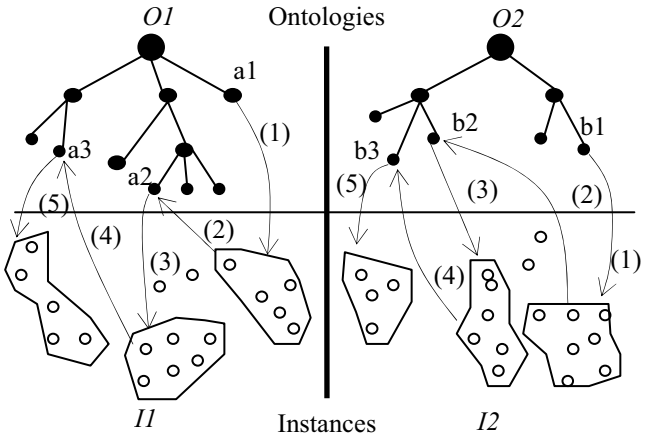


Fig1.Interleaving schema and instance matching process

This is the first transition between schema level and instance level. It is the first step in discovering instances with unique identity and indicated by arrow (1) in the figure. After discovering instances with unique identity, SBUEI utilizes these identical discovered instances and analyzes them in order to estimate similarities between concepts of schema. Similar concepts are those which have similar instances. As Fig. 1 shows, after doing resolution process between instances of *a1* and *b1* and analyzing the results, SBUEI estimates similarities between *a2* and *b2*. This is the first transition from instance level to schema level, which is represented by arrow (2). Schema matcher receives feedback from instance matcher and recognizes two equal concepts from *O1* and *O2* ontologies. After recognition of two equal concepts, SBUEI returns to instance level again (arrow (3). These processes continue consecutively until there are no instances or concepts for matching or there is not possibility for SBUEI to find more alignments. Therefore, SBUEI has two main components that are illustrated in Fig. 2: (instance matcher and schema matcher).

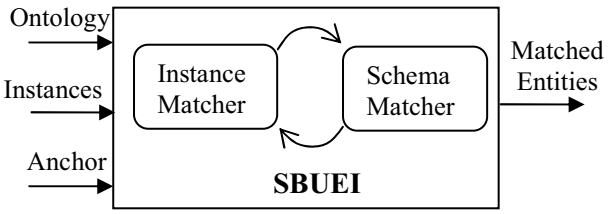


Fig2. Main Components of SBUEI

1.2 Specific techniques used

As described before, the instance coreference resolution algorithm has two phases which are executed iteratively. The first phase needs to receive an anchor as input. As the first and second phases are executed in a cycle, for the first round, the user should provide this input, but in the next times the input of the first phase (the anchors) is provided by the output of the second phase.

1.2.1 Instance coreference resolution

The instance matching process of SBUEI is completely explained in [9]. In this section, we explain the instance matching process of SBUEI concisely.

First step: create Linked Instances Cloud

We introduce a new construction that is called Linked Instances Cloud (LIC), as the basis of our instance matching algorithm.

For two equivalent concepts that we receive as input, we must create LICs. For each instance of two similar concepts, we make one LIC. If SBUEI wants to make a LIC for a specific instance, it extracts all the triples that their subjects are our intended instance and adds them to the LIC. In this way, all the neighbors of our intended instance are found. Then, SBUEI finds the triples that their subjects are instances which belong to the LIC. This means that the neighbors of the neighbors of our intended instance are found and added to the LIC. This process is actually like depth first search among neighbors of instances. SBUEI traverse across the neighbors of the instance and has a maximum depth for traversing.

The process of creating LICs is done for all of the instances of the two concepts. Creating LICs helps us in recognizing instance identities. Identities of instances are sometimes not recognizable without considering the instances that are linked to them, and neighbors often present important information about intended instances.

Second step: compute similarity between LICs and finding identical instances

In this step, the LICs of two equal concepts should be compared. Each LIC from one concept is supposed to be compared with all LICs of the other concept in order to find similar LICs. Starting points of two similar LICs, would be identical instances. For comparing two LICs, triples of two LICs should be compared. In this process, only

triples whose objects are data type values (and not instances) would participate in the comparison. Properties values are very important in comparison.

We use edit distance method and a token-based measure for comparing string values of properties. Similarity values of triples objects are added together for obtaining similarity value of two LICs. Similarities of properties values are added with a particular coefficient which has inverse relations to the depth of the subject of triples in LIC. We use a weighted sum for computing similarity of LICs. We normalize the sum of similarities of properties values in two LICs into a range of 0 and 1 and select the most similar LICs.

When two LICs are selected as two similar LICs, we consider their starting points as identical instances. In this way, some identical instances could be found regarding to their properties and their neighbors.

Third step: finding identical instances in the vicinity of identical instances

We found some identical instances with utilizing their LICs. In this step, we continue the process of matching on those LICs of the previous step that led to discovering identical instances. The strategy in this step is searching locally around the identical instances in order to find new equal instances. This means that if two instances are identical, then there is possibility that their neighbors are similar too. The process of comparing instances is similar to what mentioned in the previous steps.

1.2.2 Compute concept similarities in schema level

After finding identical instances in the neighborhood of identical instances, now it is time to find similarities between concepts in two heterogeneous schemas. In this part, instance matcher gives feedback to us for finding similar concepts in schema level. If we find some similar instances such as 'm' and 'n' in the instances of LIC_i and LIC_j (i and j are two identical instances that are detected in the second step), concepts that 'm' and 'n' belong to them would be good candidates to be similar.

The approach repeats this step for every two similar LICs and considering to identical instances in two similar LICs, estimates similarities between concepts. SBUEI used a measure in order to find a similarity value between two concepts.

The second phase is done by a schema matcher. It receives feedback from the first phase, which contains some similarities between concepts from the viewpoint of instance matcher. At this time, schema matcher begins the process of matching in schema level by applying some ontology matching algorithms. SBUEI compares all of these similarity values that are proposed by instance matcher or obtained by schema matcher, and choose a pair of concepts that have the most similarity. SBUEI repeats these two phases consecutively.

When SBUEI wants to do ontology matching, it considers to the concepts that are proposed as equal concepts in the previous iterations and the process of ontology matching starts in the neighborhood of these concepts. We applied the definition of concept neighborhood in [8]. Schema matcher utilizes two kinds of matchers: lexical matcher and structural matcher. Lexical matcher uses Princeton WordNet [10], EditDistance method and Wu-Palmer measure [11] for computing lexical similarities. In [12] structure based techniques are divided into two groups based on the internal structure and relational structure. SBUEI utilizes internal and relational structures for computing similarities between concepts.

1.4 Link to the system and parameters file

The website of SBUEI is http://nlp.sbu.ac.ir/sbuei/sbuei.html
More information about SBUEI is presented here.

1.5 Link to the set of provided alignments (in align format)

The alignments of SBUEI for OAEI campaign is available at:
http://nlp.sbu.ac.ir/sbuei/download.html

2 Results

In this section, we present the results obtained by SBUEI in the OAEI 2012 campaign. SBUEI participated in two tracks: Sandbox and IIMB. The results are evaluated in comparison with some gold standard alignments.

2.1 Sandbox Track

Sandbox is a simple data set and contains 11 test cases. Test cases contain some kinds of transformations such as data value transformation, structural transformation and logical transformation. The transformations are not as hard as the transformations of IIMB track. The data set is generated artificially. Table 1 represents the total amounts of precision, recall and F Measure for this data set.

Table 1. Sandbox Results

Test Cases	1-11
Precision	0.95
Recall	0.98
F Measure	0.96

We have encountered some reductions in precision and recall value. So, we analyzed the result and found some problems in the data set and reference alignments:

- There are some URI aliases in each test case. For example, see the URI1 and URI2 in test case 000 (test case 000 is the test case that other test cases must be matched against this test case):
  URI1: http://oaei.ontologymatching.org/2012/IIMBDATA/m/0bvgl_4
  URI2: http://oaei.ontologymatching.org/2012/IIMBDATA/m/0bvgm5l
  These two URIs depict the same identity. Both of them have exactly the same properties and values.
  On the other hand, we have some URI aliases in test case 001, such as URI3 and URI4.

URI3: http://oaei.ontologymatching.org/2012/IIMBDATA/m/item100994799229 4508239
URI4: http://oaei.ontologymatching.org/2012/IIMBDATA/m/item595676017412 1985261
URI3 and URI4 describe identical instances. Moreover, URI1, URI2, URI3 and URI4 refer to an entity and present the same identity. SBUEI found these alignments: (URI1, URI3), (URI1,URI4), (URI2,URI3), (URI2,URI4). However, only two alignments (URI1,URI4) and (URI2,URI3) belong to gold standard alignments. Therefore, our precision has decreased.

- We found an incorrect alignment in the reference alignments. See the following URI (URI5 from test case 000) which describe the English language.
  URI5: http://oaei.ontologymatching.org/2012/IIMBDATA/en/english
  In test case 001, there is an instance with the following URI (URI6) which its identity is the same as the URI5 and represents the English language.
  URI6: http://oaei.ontologymatching.org/2012/IIMBDATA/en/item720829132936 6150827
  We can find the alignment (URI5, URI6) in gold standard alignments. Nevertheless, we can also find another incorrect alignment for URI5. URI5 is matched incorrectly with an instance with URI7.
  URI7: http://oaei.ontologymatching.org/2012/IIMBDATA/en/item677301914259 3325946
  So, we have these alignments in gold standard alignments: (URI5, URI6), (URI5, URI7). But, SBUEI found only (URI5, URI6) as two identical instances. Hence, its recall has declined.

2.2 IIMB Track

IIMB data set is extracted from Freebase and includes 80 test cases. Each test case contains some kinds of different transformations. Test cases 1 to 20 contain data value transformation, 21-40 contain structural transformation, 41-60 contain logical transformation and 61-80 contain a combination of these transformations. All of these 80 test cases must be matched against a source test case. Table 2 shows the results of SBUEI on different groups of test cases (based on their transformations).

Table 2. IIMB Results

Transformations	1-20	21-40	41-60	61-80	overall
Precision	0.95	0.96	0.91	0.58	0.87
Recall	0.98	0.98	0.85	0.5	0.85
F Measure	0.97	0.98	0.87	0.53	0.86

We observed some problems in the IIMB task such as those problems that we mentioned in Sandbox task. These problems such as URI aliases have decreased our precision.

3 General comments

In this section, we provide some comments about our system and OAEI 2012 campaign.

3.1 Comments on the results

The results of our system are very promising. SBUEI obtained high value for precision, recall and F-measure in Sandbox task and test cases 1-40 of IIMB task. SBUEI has much better performance in test cases with data value transformation and structural transformation than test cases with logical transformation and combinational transformations. This means that SBUEI is very resistant to modifications such as changes in data format, removing, adding and hierarchal changing of properties. As we expected, SBUEI has its weakest performance in front of combinational transformations, and it is completely normal for systems to have weaker performance against combinational transformations than other transformation because it contains all kinds of transformation together. However, it is one of the most important shortcomings of our system and it is very beneficial to improve it by applying new techniques.

3.2 Discussions on the way to improve the proposed system

Our system participated for the first time in this competition and we focused a lot more on technical issues and our new algorithm than some usability aspects. Considering that SBUEI is a recently created approach, does not have appropriate user interface. Therefore, it is important to make a powerful user interface for SBUEI. Our future target includes utilizing some methods such as semi supervised learning algorithms to find discriminable properties in the LICs. This will help us to find similar LICs efficiently and optimize our system in order to improve some scalability aspects of our system.

3.3 Comments on the OAEI 2012 procedure

In OAEI 2012, SEALS platform is used for evaluating participating systems in all the tracks except for instance matching. It would be very beneficial for instance matching track to be run on a platform like SEALS.

3.4 Comments on the OAEI 2012 test cases

In the IIMB track of OAEI 2011, we had test cases which the size of their data sets had been heavily increased compared to the preceding years. Each test case size was more than 20MB. Therefore, participants had to deal with large data sets and their systems were evaluated considering some scalability aspects. In OAEI 2012, the sizes of data sets are not as much as the last year and they are declined. The large data sets

are more challenging for systems. Thus, it will be useful to have a better and stronger evaluation by large data sets. Moreover, we encountered some problems in reference alignments that we discussed about them in section 2. It is better to have more accurate data sets and reference alignments.

4 Conclusion

In this paper, we have described our system, SBUEI, for instance matching. SBUEI is applicable in various data sets with heterogeneous schemas. SBUEI pays attention to matching in both schema and instance level. The architecture, the main algorithms and the specific techniques of SBUEI have been presented in this report. Our experiments in Sandbox and IIMB showed that our approach achieved high precision and recall. This was the first participation of SBUEI and we obtained promising results; however, there are more technical issues that can improve the performance of SBUEI. We are going to optimize our system based on what was mentioned earlier in the future work.

References

1. Bizer, C., Heath, T. and Berners-Lee, T. Linked Data-The Story So Far, Int. J. Semantic Web Inf. Syst. 5(3), 1-22 (2009)
2. Hu, W., Chen, J. and Qu, Y. A Self-training Approach for Resolving Object Coreference Semantic Web, In: 20th International World Wide Web Conference, India (2011)
3. Noessner, J., Niepert, M., Meilicke, C., Stuckenschmidt. Leveraging Terminological Structure for Object Reconciliation, In:7th Extended Semantic Web Conference, Greece (2010)
4. Sais, F., Niraula, N., Pernelle N. and Rousset, M. LN2R a knowledge based reference reconciliation system: OAEI 2010 results, In: 5th International Workshop on Ontology Matching , China (2010)
5. Jain, P., Hitzler, P., Sheth, A. P., Verma, K. and Yeh, P. Z. Ontology Alignment for Linked Open Data, In: 9th International Semantic Web Conference, China (2010)
6. Parundekar, R. Knoblock, C. A. and Ambite, L. Linking and building of ontologies of linked data. In: 9th International Semantic Web Conference, China (2010)
7. Nikolov, A., Uren, V., Motta, E. and Roeck, A. Overcoming schema heterogeity between linked semantic repositories to improve coreference resolution, In: 4th Asian Semantic Web Conference, China (2009)
8. Seddiqui, Md. Hanif and Aono M. An Efficient and Scalable Algorithm for Segmented Alignment of Ontologies of Arbitrary Size. J. Web Sem. 7(4), 344-356.
9. Taheri, A., Shamsfard, M. Consolidation of Linked Data Resources upon Heterogeneous Schemas, In Proceedings of the Sixth International Conference on Advances in Semantic Processing (SEMAPRO 2012), Spain, September 2012.
10. Fellbaum, C. WordNet: An Electronic Lexical Database,MIT Press, Cambridge, MA (1998)
11. Wu, Z. and Palmer, M. Verb Semantics and Lexical Selection. In: 32nd Annual Meeting of the Association for Computational Linguistics. Las Cruces (1994)
12. Euzenat, J. and Shvaiko, P. Ontology Matching, 1st ed., Springer, Berlin Heidelberg (2007)

ServOMap and ServOMap-lt Results for OAEI 2012

Mouhamadou Ba[1], Gayo Diallo[1]

[1] LESIM/ISPED, Univ. Bordeaux Segalen, F-33000, France
first.last@isped.u-bordeaux2.fr

Abstract. We present the results obtained by the ontology matching tools ServOMap and ServOMap-lite within the 8th edition of the Ontology Alignment Evaluation Initiative (OAEI 2012) campaign. The mappings computation is based on Information Retrieval techniques thanks to the use of a dynamic knowledge repository tool, ServO. This is the first participation of the two systems.

1 Presentation of the systems

We describe in this paper the ServOMap system, a piece of research work related to the area of ontology matching [1]. The followed matching approach takes its roots from the Ontology Repository (OR) system ServO [2, 3] and an initial idea implemented in [4]. The ServO OR provides functionalities for managing multiple ontologies and providing indexing and searching facilities. Its design is based on the assumption that there is a real necessity to offer both the possibility of retrieving online knowledge organization systems (KOS) but also to leverage the many ad hoc thesauri and other structured vocabularies built and maintained for local purposes. Indeed, there are many KOS which are not available within the Semantic Web infrastructure and are not reachable by conventional Semantic Web search engines and repository (e.g. [5-8]). ServO offers the possibility for an automated and fast OR building for a particular application purpose. The ServoMap matching system takes benefit of ServO and is a flexible and efficient large scale ontology matching system.

1.1 Purpose and general statement

ServOMap is designed for facilitating real time interoperability between different applications which are based on heterogeneous knowledge organization systems. The heterogeneity comes from the language format, their level of formalism, etc. The system relies on Information Retrieval (IR) techniques and a dynamic description of entities of different KOS for computing the similarity between them. It is mainly designed for meeting the need of matching large scale ontologies such as [9].

From now on, if not necessary, we will mainly continue to refer to ServOMap for describing our two tools as ServOMap-lt is a version which uses only some of the settings of the system.

1.2 Techniques used

The overall followed process for matching two inputs ontologies is described in figure 1. We detail below each step.

Computing Ontology Metrics

The first step after parsing and loading input ontologies is to compute a set of metrics that are later used as parameters for the systems and for optimization purpose. These metrics include for any input ontology: the average number of child by concepts, the list of languages used to denote entities labels or their annotation properties, the most frequent single terms within the ontology, the longest set of synonyms labels used to describe a concepts.

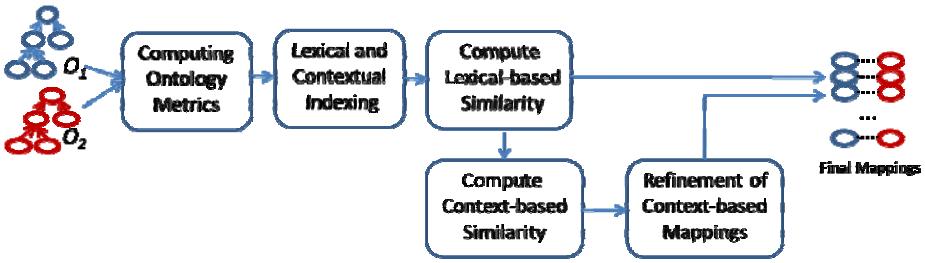


Fig. 1: ServoMap overall followed process for ontology matching

Lexical and Contextual Indexing

As ServOMap relies on IR techniques for ontology matching, an ontology is seen as a corpus of document to process where each entity (concepts, relations) is a semantic document to process.

ServOMap constructs an inverted index thanks to the use of the Ontology Indexing Module of ServO which relies on the Apache Lucene API[1]. According to the parameters computed during the previous step, a dynamic generation of each entity description is performed. This process is dynamic as each entity is described according to the features it holds. Therefore, some concepts may have synonyms in several languages or may have comments, while others may only have English terms. Moreover, some concepts may have declared properties (either object properties or data type properties), etc. During this dynamic description process, the retrieved strings from a concept are passed to a set of filters: stop words removal, normalization (upper case to lower case), punctuations removal, completion of labels by the permutations of their terms and so on. A flag is used to indicate whether ServOMap uses stemming or not and if the words of a term will be concatenated before to add them to the index. Table 1 gives an extract of available fields and their term counts within the index for the Foundational Model of Anatomy ontology (FMA). The version used for this ontology contains 79,042 entities, among them 78,884 are concepts. As we can see, the value of the ***dDomain*** field (the domain of a property) is *spatialassocirelat* which is the term "*spatial association relation*". And the concept with id *#Accessory_lobar_vein* has as ***directLabelCEn*** **(**direct label English label**)** the

[1] http://lucene.apache.org/

set {*accessorilobarvein veinaccessorilobar veinlobaraccessori*} for "Accessory lobar vein" and its permutations**.** All spaces are removed between words.

Field Name	Term Counts	Example
dDomain	15	spatialassocirelat
dRange	5	string
directLabelCEn	152,088	accessorilobarvein veinaccessorilobar veinlobaraccessori
directNameC	78,884	accessorilobarvein
directNameP	52	percentag
uri	79,042	http://bioontology.org/#Accessory_lobar_vein

Table 1: An extract of an entry index for the Mouse Anatomy Ontology

Compute lexical based similarity

After the indexing phase, ServOMap proceeds to the computing of lexical based similarity. This step relies on the Ontology Retrieval Module of the ServO OR.

Depending on the flag indicating the indexed ontologies, the Ontology Processing Module is called for retrieving the concepts to use for searching over the built index. Thus, if both input ontologies are indexed, the first one, let's say O_1, is used as search ontology over the index on the second ontology I_2. And, vice versa, the ontology O_2 is used to perform search over the index of the first ontology I_1. If the flag indicates that one ontology is indexed, then ServOMap performs only a one way search.

As in the lexical and contextual indexing phase, a dynamic generation of entity description if performed for any entity to use in order to search the index. A Boolean query is constructed with all the available fields for the entity. Each Boolean query, represented as a vector of terms, is searched over the index. A ranked list of entities is retrieved. ServOMap keeps the result constituted by the couple of the entity to search and the entity having the highest similarity as a possible mapping (vectorial similarity). It can happen that several entities have the same similarity with the entity to search. In this case, in order to keep the most relevant one, the names of the entities are compared using the Levenshtein Distance.

Compute context-based similarity

The idea of context-based similarity is based on the assumption that when two entities are similar, there is a big chance that the concepts that surround it are also similar. Here, by surrounding concepts (context) we mean super-concepts, sub-concepts and siblings concepts. Therefore, in the context based similarity, the description of a concept is based on its context. This context based similarity is

applied only on concepts and not on the properties of the ontologies to match. In addition, we restrict the contextual similarity computing to only the concepts that have not been yet mapped to any other concepts by the lexical-based similarity. This is based on the assumption that if two concepts are mapped by the previous lexical strategy, it is likely to be correct.

Refining mappings obtained from context based similarity

The mappings with context similarity are less accurate. The idea is thus to avoid keeping a couple obtained from the context based similarity where one of the entries is already mapped during the lexical process by another concept. This strategy takes into account the worst case and allows removing several incorrect mappings and increase the recall at the same time. However, it generates false positive correspondences, and the precision obtained with lexical-based mappings is then reduced.

Processing disjoints concepts

For ontology matching, some inputs ontologies are described with complex axioms. In particular, it is possible to have disjointness statements. In such a case, we use an algorithm for processing these particular issues. Let's assume that C_1 and C_2 are two disjoints concepts belonging to an ontology O_1 and C_3 and C_4 two other disjoints concepts belonging to the ontology O_2. During the indexing phase, we complete the description of C_1 by adding a field for its disjoint concepts and the same for C_2, etc. These information is later used to avoid let's say mapping both $C_1 - C_3$ and $C_1 - C_4$.

	ServOMAP	**ServOMap-lt**
Terms processing	According to the language of the labels	The same for all languages
Entities taken into account	All	Only Classes
Ontologies indexed	Both	One
Searching strategy	Two ways	One way
Stemming	No	Yes
Arity	1:1	1:n

Table 2: Configurations of ServOMap and ServOMap-lt

1.3 Adaptations made for the evaluation

The ServO OR system uses a threshold as parameter for possibly limiting the retrieved concepts from the index. For ServOMap we limited the results to the best similarity.

Our system participated to the campaign with two versions of our approach corresponding to different parameters settings. The main differences in term of parameters are presented in table 2.

In addition to these parameters, we used only the first step of similarity computing. And our system does not use a particular knowledge background.

1.4 Link to the system and parameters file

The Seals wrapped ServoMap and ServOMap tools are available online at http://code.google.com/p/servo/.

2 Results

In this section, we provide comments on the official results obtained by the two configurations of the ServOMap matching system.

2.1 benchmark

The Benchmark track 2012 includes 111 tests. Each test concerns a source ontology called reference and a test ontology which is created by modifying some information from the reference alignment. For the provided dataset (finance, bench2, bench3, bench 4 and biblio) ServOMap performed better than ServOMap-lt thanks to the better recall. Due to the one way searching strategy of ServOMap-lt, it is faster but its configuration based on stemming and only classes-based strategy reduced its F-measure.

2.2 anatomy

The precision of our system are very good on the Anatomy track where the ServOMap configuration provided the best precise mappings (0.996). In term of computation times, ServoMap-lt completed the task in less than 25 seconds.

2.3 conference

For the conference track, contrary to the results obtained using directly the Seals Plateform, the official provided results were filtered out by removing all instance-to-any_entity and owl:Thing-to-any_entity correspondences prior to computing

Precision/Recall/F1-measure. Our system was able completing the 120 alignments in 64 seconds for the ServOMap configuration and in 51 seconds for SevOMap-lt.

2.4 multifarm

Even if our system is able to deal with multilingual ontologies, the cross-lingual ontology mapping has not yet been implemented, which is the case with the multifarm task. We were able processing the inputs ontologies but fail computing correct mappings at this time.

2.5 library

The library track is about matching two thesauri, the STW and the TheSoz thesaurus. They provide a vocabulary for economic respectively social science subjects and are used by libraries for indexation and retrieval. As our ontology processing module relies on the Jena Framework [10], we experienced an issue processing the input ontologies because of their formatting. However, we were eventually able completing the task and correctly handled multilingual terminologies associated with the entities in these KOS. ServOMap-lt and ServOMap were among the best systems, ranked second and third respectively in term of F-measure (0.670 and 0.665). ServOMap finished the task in 44 seconds (second) and ServOMap-lt in 45 seconds.

2.6 large biomedical ontologies

Our tool in both configurations was able completing the large biomed track (LargeBio), which was the most challenging one regarding particularly the number of entities involved in the matching task. We found the NCI thesaurus very time consuming for context based mapping as its concepts have many siblings. Table 3 summarizes the performances obtained by the ServOMap and ServOMap-lt on the LargeBio track. ServOMap provided overall the best precision mappings among all the participating systems (0.903) and completed all the tasks in 2,310 seconds. ServOMap-lt was ranked second in term of F-measure with 0.780 and completed all the tasks in 2,405 seconds.

	ServOMap				**ServOMap-lt**			
	P	**R**	**F**	**T (s)**	**P**	**R**	**F**	**T(s)**
FMA-NCI	0.945	0.747	0.834	327	0.931	0.8	0.86	366
FMA-SNOMED	0.953	0.656	0.777	893	0.956	0.60	0.802	790
SNOMED-NCI	0.901	0.554	0.687	1,089	0.875	0.593	0.706	1,248

Table 3: Performance obtained on the 2012 LargeBio track

3 General comments

3.1 Comments on the results

Our system performs well for knowledge organization systems having concepts described by several synonyms terms regardless their languages as it depends heavily on the lexical description of the resources. However, for the tasks which relies more on the structural description of ontologies, our system performs less. Overall, the precision is very good, in particular for the ServOMap configuration as its uses a very discriminating strategy during the search process (two ways search).

3.2 Discussions on the way to improve the proposed system

So far our system is not using any external resources apart from the usual stops words list constituted by the common terms discarded during indexing and searching. It relies only on the intrinsic information encoded into the input ontologies. Our system could be improved then by the use of external resources for instance for morphological and lexical variation of terms or by the use of the UMLS and its semantic network for removing incorrect mappings found during the context-based similarity. In addition, completing the lexical and contextual description of entities by *true* structural information could also improve the results. Also, as ServOMap is not able to compute oriented mapping, which is quite challenging with an approach relying on the lexical description of entities, structural description could help. From computation time point of view, implementing multithreading can be a possible way to improve the system.

3.3 Comments on the OAEI 2012 procedure

As a first participation, we found the OAEI procedure very convenient and the organizers very supportive. The use of Seals allows objective assessments.

3.4 Comments on the OAEI 2012 test cases

The OAEI test cases are various and this leads to comparison on different levels of difficulty, which is very interesting. In addition, real world ontologies are provided.

4 Conclusion

This 2012 edition of OAEI is our first participation in the campaign. The results obtained both by ServOMap and ServOMap-lt are quite very promising both for F-measure and computing times. The version of our system which uses the whole

configuration performed less than the lite one on the Large Biomed task in term of F-measure while it gives the best precision. The lite version is less stable regarding the others tasks.

Our ontology matching system presents some limitations. And there is a room of improvements. First, we plan to improve the algorithm used for filtering out the mappings provided by the context-based matching in order to increase the recall without reducing the precision. Also, ServOMap does not use any external resource in the similarity computing process. We intend to use the UMLS resource for better discarding incorrect mappings for life sciences related ontologies. Moreover, the current version does not provide oriented mapping nor takes into account matching two ontologies described in two different languages (e.g. English Vs French). Thus, an improvement of the system is the implementation of a cross lingual ontology matching approach and investigating into oriented mappings issue. Finally, we plan introducing logic assessment of computed mappings [11] and implementing a user friendly interface.

References

1. Euzenat, J, Meilicke, C, Stuckenschmidt, H, Shvaiko, P, Trojahn, C.: Ontology Alignment Evaluation Initiative: six years of experience. J Data Semantics (2011)
2. Diallo G. Efficient Building of Local Repository of Distributed Ontologies. In Proceedings of International Conference on Signal-Image Technology and Internet Based Systems - SITIS'2012, pp. 159–166. IEEE
3. Diallo G. Towards decentralized and cooperative repositories of distributed ontologies. In Proceedings of SWAT4LS 2011, pp. 8–9
4. Diallo G, Simonet M, Simonet A. Bringing Together Structured and Unstructured Sources: The OUMSUIS Approach. OTM Workshops (1) 2006: 699-709
5. Ding, L, Finin, T, Joshi, A, Pan, R, Cost, RS, Peng, Y, Reddivari, P, Doshi, V, Sachs, J (2004). Swoogle : a search and metadata engine for the semantic web. In Proceedings of the Thirteenth ACM Conference on Information and Knowledge Management. pages 652-659.
6. Côté RG, Jones P, Apweiler R, Hermjakob H. The Ontology Lookup Service, a lightweight cross-platform tool for controlled vocabulary queries. BMC Bioinformatics. 2006;7:97
7. d'Aquin, M, Baldassarre, C, Gridinoc, L, Angeletou, S, Sabou, M, Motta, E. Watson: A Gateway for Next Generation Semantic Web Applications. Poster session of the International Semantic Web Conference, ISWC 2007.
8. Noy NF, Shah NH, Whetzel PL, Dai B, Dorf M, Griffith N, et al. BioPortal: ontologies and integrated data resources at the click of a mouse. Nucleic Acids Research. 2009 May 29;37
9. Ruiz EJ, Grau BC, Zhou Y, Horrocks I. Large-scale Interactive Ontology Matching: Algorithms and Implementation. Proceedings of the 20th European Conference on Artificial Intelligence (ECAI). IOS Press; 2012. p. 444–9
10. Carroll, JJ, Dickinson, I, Dollin, C, Reynolds, D, Seaborne, A, Wilkinson, K. Jena: implementing the semantic web recommendations. In Proceedings of the 13th International World Wide Web Conference, pp. 74-83, New York (2004)
11. Meilicke C, Stuckenschmidt H, Sváb-Zamazal O. A Reasoning-Based Support Tool for Ontology Mapping Evaluation. ESWC. 2009. p. 878–82

TOAST Results for OAEI 2012

Arkadiusz Jachnik, Andrzej Szwabe, Pawel Misiorek, and Przemyslaw Walkowiak

Institute of Control and Information Engineering, Poznan University of Technology,
M. Sklodowskiej-Curie Square 5, 60-965 Poznan, Poland
{arkadiusz.jachnik,andrzej.szwabe,pawel.misiorek,przemyslaw.walkowiak}@put.poznan.pl

Abstract. The Tensor-based Ontology Alignment SysTem (TOAST) is a general-purpose (i.e., domain-unspecific) self-configurable (i.e., requiring no user intervention) ontology matching tool. TOAST is based on one of the first tensor-based approaches to Statistical Relational Learning. Being one of the possible applications of the Statistical Relational Learning framework, TOAST may be seen as a system realizing a probabilistic inference with regard to a single relation only - the relation representing the 'semantic equivalence' of ontology classes or their properties. Due to the flexibility of the integrated tensor-based representation of heterogeneous data, TOAST is able to learn the semantics equivalence relation on the basis of partial matches data included in a train set.

1 Presentation of the System

The Tensor-based Ontology Alignment SysTem (TOAST) presented in this paper is an application of an extended version of our tensor-based approach to Statistical Relational Learning (SRL) referred to as Tensor-based Reflective Relational Learning Framework (TRRLF) [12]. In general, SRL is one of the most intensively investigated problems of Artificial Intelligence. Recently proposed tensor-based SRL methods are widely regarded (e.g., see [9]) as a promising alternative to the commonly used graphical models, such as Bayesian Networks and Markov Logic Networks [2], [5]. To our knowledge, TOAST represents the first tensor-based approach to ontology alignment.

We use a 3rd-order tensor as a data structure that is suitable to represent data provided as a set of RDF triples [4], [9]. There are several recent works considering the use of tensors to represent relational data given as RDF triples [5], [9], [4], [11]. The authors of these works assume that the *active* mode (corresponding to the RDF subject role) and the *passive* mode (corresponding to the RDF object role) of each entity have to be modeled as two separate tensor modes. However, they do not address the questions of (i) how to model the relation between two modes of the same entity and (ii) how the orientation of this relation (i.e., the setting which entity plays the active and which entity plays the passive role, as far as a given relation is concerned) influences the system performance [9], [4], [11].

We intend to confront these issues by proposing to model data in a way that enables a high level of flexibility for specifying the roles that any pair of entities plays with regard to any relation. Consequently, we represent both the *active* and *passive* modes of a given entity as potentially fully independent of each other – it is the correlation of the

active mode and the passive mode (observable in the input data) that fully determines the extent to which the vectors representing the modes are algebraically similar to each other.

As we have shown in our experiments, the proposed tensor-based representation of relational data (in particular RDF triples), is appropriate for the ontology alignment task. It is worth noting that the internal data representation of TOAST is based on a probabilistic model of a vector space that has so far only been used in quantum Information Retrieval [13].

It should be stressed that TOAST does not require the use of external knowledge sources, such as dictionaries or thesauruses, in order to provide high quality results. However, the use of such knowledge data is possible – it may be realized by converting the data into the subject-predicate-object format [12], as discussed in Section 3.

1.1 State, Purpose, General Statement

TOAST is a fairly general-purpose ontology alignment tool. Being a specialized application of our SRL framework (i.e., the TRRL framework), TOAST may be seen as a system realizing a probabilistic inference with regard to a single relation only - the relation representing the semantic equivalence of ontology classes or their properties. The TRRL's flexibility, which is typical of SRL methods, is clearly visible in the propositional representation of all the heterogeneous data provided to the system (including the propositional representation of the occurrence of terms in the labels of the ontology classes).

The evaluation of TOAST has focused on the Anatomy track, which belongs to OAEI tracks that involve the use of the most expressive ontologies [3]. For this reason, we have not prepared the TOAST system to parse input data for any OAEI track other than the Anatomy. As a result, the Anatomy test is the only OAEI track test that TOAST passes. On the other hand, it should be noted that, in 2012, TOAST is the only matching system that can exploit additional partial alignments in the Anatomy track. To illustrate this fact, we present an additional experimental evaluation that has been performed, as suggested by the OAEI organizers, with the use of the OAEI 2010 dataset[1], in case of which the train set includes partial alignments. We show that, when partial alignments are available, TOAST is able to learn the semantics of all the relations [8], including the *matchesTo* relation, on the basis of the partial alignments data. It allows the system to exploit 'a behavioral dimension' of the alignments modeling and generation [12].

The results of TOAST evaluation presented in this paper are comparable with the results of the leading systems that have been evaluated from the perspective of Subtask #4 of the 2010 Anatomy track edition[2].

[1] Anatomy 2010 modified dataset: `http://oaei.ontologymatching.org/2010/anatomy/modifications2010.html`

[2] Anatomy - Results of 2010 Evaluation: `http://oaei.ontologymatching.org/2010/results/anatomy/index.html`

1.2 Specific Techniques Used

As TOAST is based on an SRL method, all techniques that are used in the system may be regarded as SRL solutions, rather than solutions specific to the ontology matching task. From such a general perspective, TOAST may be seen as a system that exploits a new algebraic data representation and processing method as a means for ontology alignment.

Tensor-Based Relational Data Representation

The tensor used in TOAST [12] can be seen as tensor product $T_{i,j,k} = [t_{i,j,k}]_{n \times n \times m} = S \times O \times R$ of vector spaces whose coordinates correspond to the set of subjects S, the set of objects O, and the set of relations R. We assume that $|R| = m$ and that $|S| = |O| = n$. Additionally, we define set F as a set of all the known facts (i.e., RDF triples) which are used to build the input tensor. The number $|F| = f$ determines the number of positive cells in the input tensor. Moreover, we define set $E = S \cup O \cup R$ as a set of elements (i.e., subjects, objects, and relations) used in the input data and represented in T by a slice (2nd-order array) of the 3rd-order tensor [12]. Due to the flexibility of the proposed tensor data model, it is possible to integrate the information about the ontology schema structure with the lexical knowledge. Therefore, set F contains facts about the relations between the ontology entities as well as between the ontology entities and the terms (representing lexical information) [12].

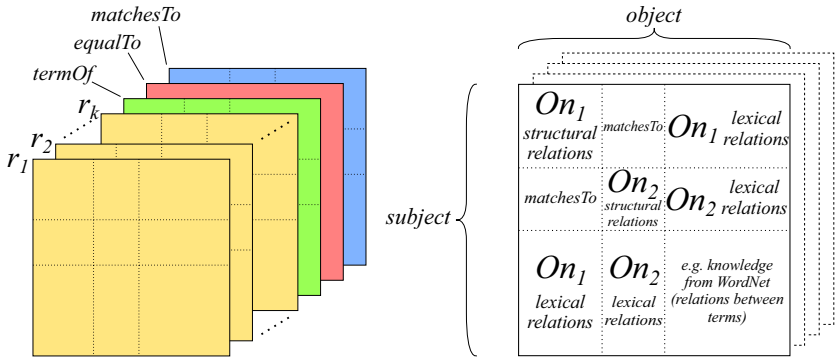


Fig. 1. The TRRLF tensor model and the TOAST tensor slice model.

Each tensor slice merges several submatrices and may be interpreted as a block matrix, as illustrated in Figure 1. For the case of a slice with structural information (i.e., representing *subClassOf* or *partOf* relations), the two submatrices on the diagonal represent a given relation for source ontology On_1 and target ontology On_2, respectively. Lexical relation slices (e.g., *termOf* slices) contain term-node submatrices describing the occurrences of terms in labels of the ontology classes. It should be stressed that TOAST allows us to use additional knowledge in the form of *partial reference alignments*. These partial alignments are represented by entries of an additional slice. Each

of these entries represents the extent of the *matchesTo* relation between a given pair of nodes.

Common Vector Space

TRRLF uses a common d-dimensional vector space [12] to represent *context vectors* for all subjects from S, objects from O, and relations from R, as well as for all facts from F stored in the input tensor T. The context vectors set is modelled as matrix $X = [x_{i,j}]_{(2n+m+f)\times d}$, where:

$$X_{(2n+m+f)\times d} = \begin{bmatrix} X^E_{(2n+m)\times d} \\ X^F_{f\times d} \end{bmatrix}, \text{ where } X^E_{(2n+m)\times d} = \begin{bmatrix} X^S_{n\times d} \\ X^O_{n\times d} \\ X^R_{m\times d} \end{bmatrix}. \quad (1)$$

The matrices X^E and X^F store context vectors of the elements from E and facts from F, respectively. X^E consists of three submatrices X^S, X^O, and X^R. The initial form of matrix X is prepared with the use of the random indexing procedure which ensures that non-zero values are uniformly distributed [1].

Learning and Matching Generation

In our approach, the learning procedure is based on the updating of the *context vectors*. The procedure is executed in steps, called *reflections*. A given reflection involves the reflective data processing [12], which is similar to Reflective Random Indexing [1]. As a result of modeling predicates as contex vectors, the system is able to process multi-relational data.

We introduce matrix $A = [a_{i,j}]_{(2n+m)\times f}$ as the source of data used in the learning process. Matrix A is constructed as a result of the 'flattening' operation applied to a tensor through all three dimensions (modes) [12].

The learning process consists of consecutive reflections. Each reflection consists of the training step (i.e., the context vector update based on learning matrix A) and the normalization step (based on the 3-norm) [12]. The method involves the application of the entropy-based criterion to indicate the optimal number of reflections. The description of this criterion is beyond the scope of this paper.

The matching likelihood prediction procedure is based on the use of the 1-norm of the Hadamard product of three vectors from X: vector $x^S_{i,\cdot}$ which corresponds to the ontology class in the subject mode, vector $x^O_{j,\cdot}$ which corresponds to the ontology class in the object mode and $x^R_{k,\cdot}$ which corresponds to the *matchesTo* relation. More formally, the probability that a match exists between the entities of the input ontologies is calculated according to the following formula:

$$p_{i,j,k} = \left\| x^S_{i,\cdot} \circ x^O_{j,\cdot} \circ x^R_{k,\cdot} \right\|_1.$$

1.3 Adaptations Made for the OAEI Evaluation

For the OAEI Anatomy track evaluation, the TOAST input tensor has been generated using the following information extracted from the two input ontologies and partial reference alignments:

- structural information represented by relations *subClassOf* and *partOf*,
- lexical information represented by relation *hasTerm* and its inversion *termOf* (as explained above, we use both *hasTerm* and *termOf* relations, in order to avoid imposing an arbitrary direction of the lexical relation),
- lexical information represented by two additional slices built on the basis of `oboInOwl:hasRelatedSynonym` and `oboInOwl:hasDefinition`,
- additional partial reference alignments (i.e., the *matchesTo* relation) represented by an additional slice.

1.4 Link to the System and Parameters File

The TOAST system is available at `www.cie.put.poznan.pl/toast/TOAST_2012.zip`. The TOAST alignments (in the RDF alignment format) together with the configuration files for OAEI 2012 and OAEI 2010 may be found at `www.cie.put.poznan.pl/toast/results2012.zip`.

2 Results

In this section, we present the results of the evaluation of TOAST performed as part of the OAEI 2012 campaign. We have participated only in the Anatomy track of OAEI 2012. This year TOAST has been identified as the only matching tool evaluated in OAEI that is able to exploit partial alignments of the Anatomy track. Unfortunately, for this reason the organizers have dropped this specific type of evaluation. Nevertheless, we have decided to show that our system is able to effectively use the additional partial alignments from the OAEI 2010 edition dataset (see Subtask #4 in the OAEI 2010 edition).

The official OAEI evaluation procedure has been executed on an Ubuntu machine with 2-core x64 processor and 4GB RAM. We additionally present the results obtained when using our machine with Ubuntu OS, 4-core processor and 16GB RAM.

2.1 Anatomy 2012 Track

OAEI 2012 Evaluation Table 1 gathers the results of the TOAST system evaluation expressed in terms of precision (P), recall (R), the F_1 measure, the number of returned matches (RM), true positives (TP), false positives (FP), false negatives (FN), trivial true positives (TP-trivial), and non-trivial true positives (TP-non-trivial). Two experiments have been executed: one by the organizers of the OAEI campaign and the other by the authors.

Table 1. The results of TOAST evaluation in the Anatomy 2012 track.

No.	TOAST config.:	P	R	F_1	RM	TP	FP	FN	TP-trivial	TP-non-trivial	time $[s]$
1	OAEI official evaluation	0.854	0.755	0.801	*no data available*						3464[1]
2	Auto-config	0.852	0.749	0.797	1333	1136	197	380	914	222	1218[2]

[1] execution time on the OAEI organizers' machine,
[2] execution time on the authors' machine.

Anatomy 2010 with Additional Partial Alignments Table 2 presents the results of our system application in the Anatomy Subtask #4 track involving the use of partial alignments. The experiments show TOAST operating in its default, fully automatic mode. Besides using the additional knowledge derived from partial alignments, we have also used information about synonyms embedded in the ontologies (relation `oboInOwl:hasRelatedSynonym`). This information has been stored as an additional tensor slice with the lexical data.

Table 2. The results of TOAST evaluation for the Anatomy 2010 Subtask #4 dataset.

No.	TOAST config.:	P	R	F_1	RM	TP	FP	FN	TP-trivial	TP-non-trivial	time[1] $[s]$
3	Auto-config	0.885	0.776	0.827	1332	1179	153	341	930	249	1941
4	Auto-config + synonyms	0.908	0.789	0.844	1320	1199	121	321	932	267	2476

[1] execution time measured on the authors' machine.

2.2 Benchmark, Conference, Multifarm, Library, Large Biomedical Ontologies, and Instance Matching tracks

As already mentioned above, in our research on TOAST, we have focused only on the Anatomy track. For this reason, we have not prepared our tool to parse input data for any OAEI track other than Anatomy, namely Benchmark, Conference, Multifarm, Library, Large Biomedical Ontologies, and Instance Matching tracks.

3 General Comments

In the following section, we provide the general comments about the TOAST results and future improvements as well as our suggestions concerning possible directions of the OAEI contest enhancements.

3.1 Comments on the Results

In the OAEI Anatomy 2012 evaluation, the self-configuring variant of TOAST achieves a comparatively high quality (see Table 1). The slight difference between our results

and the results obtained by the organizers is due to the application of slightly different techniques for computing precision and recall. In our experiment, we have used the standard method that is featured by the SEALS client.

On the basis of the results of the OAEI Anatomy 2010 evaluation, it may be additionally demonstrated that the availability of partial alignments allows TOAST to improve the matches quality. Moreover, evaluation 4 (see Table 2) has revealed the importance of additional lexical knowledge (synonyms) for improving the quality of TOAST-generated mappings.

3.2 Discussions on the Way to Improve the Proposed System

The TOAST version prepared for OAEI does not use any domain-specific background knowledge sources (such as biomedical ontologies). However, the relational data representation and processing capabilities of TOAST enable the system to exploit any generic knowledge source or linguistic resource such as WordNet [10]. In the case of any SRL-based ontology matching system (such as TOAST), taking the advantage of using external data sources (especially sources of structured data) is comparatively easy.

3.3 Comments on the OAEI Anatomy Dataset with the Partial Alignments

We have performed a lexical analysis of the Anatomy dataset and have identified several subsets of different types of alignments present in this dataset. As a result of this analysis, it has been established that the set of reference alignments contains 933 *trivial* matches (i.e., literal matches that can be found by simple string comparison) and 587 *non-trivial* matches (that require more sophisticated analysis to be identified), while the set of partial matches consists of 928 *trivial* matches and only 59 *non-trivial* matches.

This shows that the set of partial alignments contains $\sim 65\%$ of the reference matches set. However, the analogical proportion of the *trivial* and *non-trivial* matches numbers differs greatly. It can be concluded that Anatomy 2010 dataset including partial alignments is rather poorly balanced, which makes it unsuitable for a reliable evaluation in relevance feedback scenarios. Following the methodology widely used in the field of Information Retrieval [6], we suggest the development of a new subset of partial matches randomly chosen from the set of reference matches. We believe that such a dataset modification will help to increase the interest in Anatomy Subtask #4, which is the only OAEI scenario that deals with the use of relevance feedback data.

3.4 Proposed New Measures

We suggest the extension of the evaluation measures set by the Area Underneath an ROC curve - AUROC measure [6]. AUROC is a widely-used probabilistically interpretable classification quality measure. Although using AUROC requires the availability of data on incorrect matches, unknown mappings do not influence the AUROC measurement result. AUROC is regarded as the best recommendation quality measure, as long as one assumes that the purpose of an evaluated recommender is to sort all items according to their estimated usefulness. Therefore, the AUROC results may enrich the

OAEI evaluation by enabling not only the examination of the matching results given as a set, but also the evaluation of the order of the matches generated by means of the evaluated systems.

4 Conclusions

TOAST, as an application of the general-purpose Tensor-based Reflective Relational Learning framework, may be regarded as a universal (i.e., domain-unspecific) ontology matching tool. To our knowledge, TOAST is the first tensor-based approach to ontology alignment that integrates the structural and lexical data in a relational way. We have shown that the system is self-configurable and provides high-quality results. Moreover, the tool is able to effectively use partial matches data.

Acknowledgments. This work is supported by the Polish Ministry of Science and Higher Education under grant N N516 196737, and by Poznan University of Technology under grant DS 45-085/12.

References

1. Cohen, T., Schaneveldt, R., Widdows, D.: Reflective Random Indexing and Indirect Inference: A Scalable Method for Discovery of Implicit Connections. Journal of Biomedical Informatics 43(2), 240256 (2010)
2. De Raedt, L.: Logical and Relational Learning. Springer (2008)
3. Euzenat, J., Ferrara, A., van Hage, W. R., Hollink, L., Meilicke, C., Nikolov, A., Ritze, D., Scharffe, F., Shvaiko, P., Stuckenschmidt, H., Svb-Zamazal, O., dos Santos, C.T.: Results of the ontology alignment evaluation initiative 2011. In: OM-2011 (2011)
4. Franz, T., Schultz, A., Sizov, S., Staab, S.: Triplerank: Ranking Semantic Web Data by Tensor Decomposition. In: The Semantic Web-ISWC 2009, pp. 213–228 (2009)
5. Getoor, L., Taskar, B.: Introduction to Statistical Relational Learning. MIT Press (2007)
6. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating Collaborative Filtering Recommender Systems. ACM Trans. Information Systems, 22(1), 5–53 (2004)
7. Kolda, T. G., and Bader, B. W. Tensor Decompositions and Applications, SIAM Review 51(3):455-500 (2009)
8. Lavrenko V., A Generative Theory of Relevance. Springer-Verlag, Berlin (2010)
9. Nickel, M., Tresp, V., Kriegel, H.P.: A Three-Way Model for Collective Learning on Multi-Relational Data. In Proceedings of the 28th International Conference on Machine Learning, pp.809-816 (2011)
10. Princeton University. WordNet - A lexical database for English. Online: `http://wordnet.princeton.edu`
11. Sutskever, I., Salakhutdinov, R., Tenenbaum, J. B.: Modelling Relational Data Using Bayesian Clustered Tensor Factorization. Advances in Neural Information Processing Systems, 22 (2009)
12. Szwabe, A., Misiorek, P., Walkowiak, P.: Tensor-based Relational Learning for Ontology Matching. In Grana M. et al. (eds.) Advances in Knowledge-Based and Intelligent Information and Engineering Systems (KES2012), Frontiers in Artificial Intelligence and Applications vol. 243, pp. 509-518. IOS Press (2012)
13. van Rijsbergen, C. J.: The Geometry of Information Retrieval. Cambridge University Press. New York, USA (2004)

WeSeE-Match Results for OEAI 2012

Heiko Paulheim

Technische Universität Darmstadt
paulheim@ke.tu-darmstadt.de

Abstract. WeSeE-Match is a simple, element-based ontology matching tool. Its basic technique is invoking a web search engine request for each concept and determining element similarity based on the similarity of the search results obtained. Multi-lingual ontologies are translated using a standard web based translation service. The results show that the approach, despite its simplicity, is competitive with the state of the art.

1 Presentation of the system

1.1 State, Purpose, and General Statement

The idea of *WeSeE-Match* is to use information on the web for matching ontologies. When developing the algorithm, we were guided by the way a human would possibly solve a matching task. Consider the following example from the OAEI anatomy track[1]: one element in the reference alignment are the two classes with labels *eyelid tarsus* and *tarsal plate*, respectively. As a person not trained in anatomy, one might assume that they have something in common, but one could not tell without doubt.

For a human, the most straight forward strategy in the internet age would be to search for both terms with a search engine, look at the results, and try to figure out whether the websites returned by both searches talk about the same thing. Implicitly, what a human does is identifying relevant sources of information on the web, and analyzing their contents for similarity with respect to the search term given. This naive algorithm is implemented in *WeSeE-Match*.

1.2 Specific Techniques Used

The core idea of our approach is to use a web search engine for retrieving web documents that a relevant for concepts in the ontologies to match. For getting search terms from ontology concepts (i.e., classes and properties), we use the labels, comments, and URI fragments of those concepts as search terms. The search results of all concepts are then compared to each other. The more similar the search results are, the higher the concepts' similarity score.

To search for websites, we use the Microsoft Bing Search API[2]. We use URI fragments, labels, and comments of each concept as search strings, and perform some preprocessing, i.e., splitting camel case and underscore separated words into single words,

[1] http://oaei.ontologymatching.org/2012/anatomy/
[2] http://www.bing.com/toolbox/bingdeveloper/

and omitting stop words. While the approach itself is independent of the actual search engine used (although the results might differ), we have chosen Bing to evaluate our approach because of the larger amount of queries that can be posed in the free version (compared to, e.g., Google).

For every search result, all the titles and summaries of web pages provided by the search engine are put together into one *describing document*. This approach allows us to parse only the search engine's answer, while avoiding the computational burden of retrieving and parsing all websites in the result sets. The answer provided by the Bing search engine contains titles and excerpts from the website (i.e., some sentences surrounding the occurance of the search term in the website). Therefore, we do not use *whole* websites, but ideally only *relevant parts* of those web sites, i.e., we exploit the search engine both for information retrieval and for information extraction.

For each concept c, we perform a single search each for the fragment, the label, and the comment (if present), thus, we generate up to three documents $doc_{fragment}(c)$, $doc_{label}(c)$, and $doc_{comment}(c)$. The similarity score for each pair of concepts is then computed as the maximum similarity over all of the documents generated for those concepts:

$$sim(c_1, c_2) := max_{i,j \in \{fragment,label,comment\}} sim^*(doc_i(c_1), doc_j(c_2)) \quad (1)$$

For computing the similarity sim^* of two documents, we compute a TF-IDF score, based on the complete set of documents retrieved for all concepts in both ontologies.

Using the TF-IDF measure for computing the similarity of the documents has several advantages. First, stop words like *and*, *or*, and so on are inherently filtered, because they occur in the majority of documents. Second, terms that are common in the domain and thus have little value for disambiguating mappings are also weighted lower. For example, the word *anatomy* will occur quite frequently in the anatomy track, thus, it has only little value for determining mappings there. On the other hand, in the library track, it will be a useful topic identifier and thus be helpful to identify mappings. The TF-IDF measure guarantees that the word *anatomy* gets weighted accordingly in each track.

The result is a score matrix with elements between 0 and 1 for each pair of concepts from both ontologies. For each row and each column where there is a score exceeding τ, we return that pair of concepts with the highest score as a mapping. Since most ontology matching problems only look for $1:1$ mappings, we optionally use edit distance for tie breaking if there is more than one candidate sharing the maximum score. This happens, for example, for pairs like *Proceedings – Proceedings* and *Proceedings – InProceedings* in the conference track, which get very similar scores. Using the edit distance as a mechanism for tie breaking ensures that *Proceedings* is mapped to *Proceedings* and not to *InProceedings*.

Figure 1 shows the entire process using the introductory example from the OAEI anatomy dataset, computing the similarity score for *tarsal plate* and *eyelid tarsus*.

For multi-lingual ontologies, we first translate the fragments, labels, and comments to English as a pivot language [2], using the Bing Search API's translation capabilities. The translated concepts are then processed as described above. The whole process is illustrated in Fig. 2.

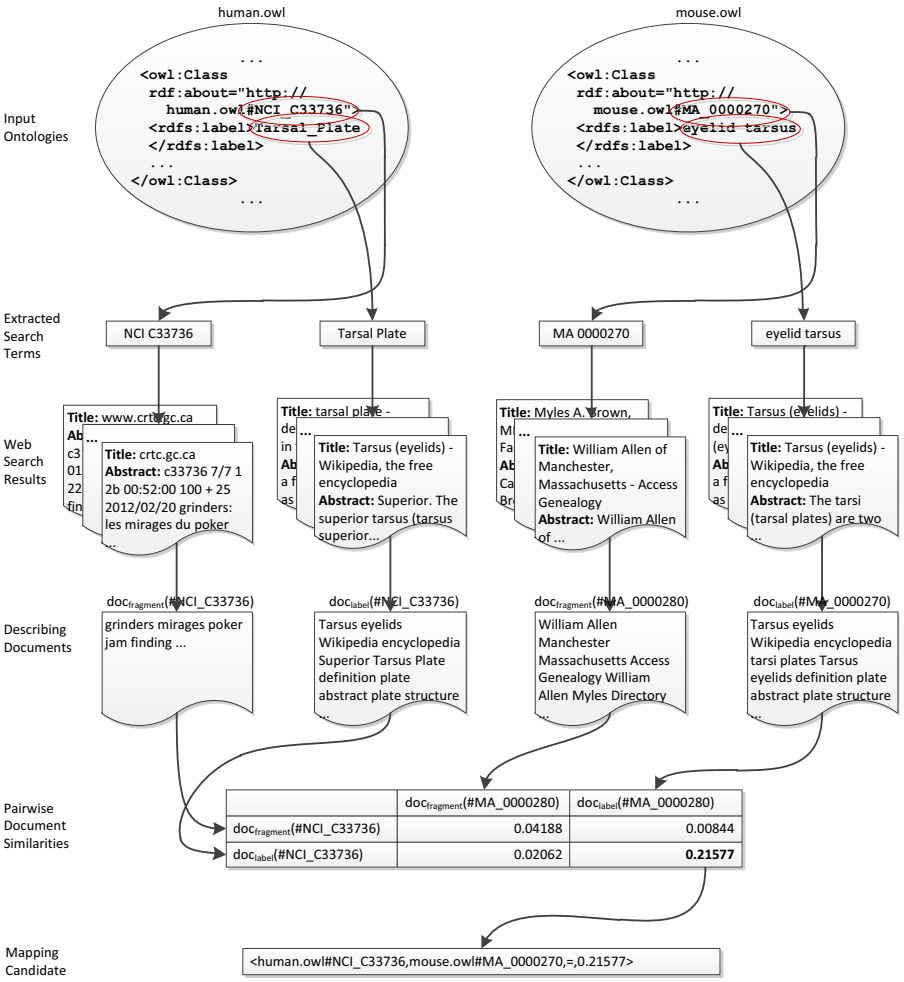


Fig. 1. Example with two concepts from the OAEI anatomy dataset. This is a mono-lingual case; for multi-lingual ontologies, an additional translation step is performed on the extracted search terms.

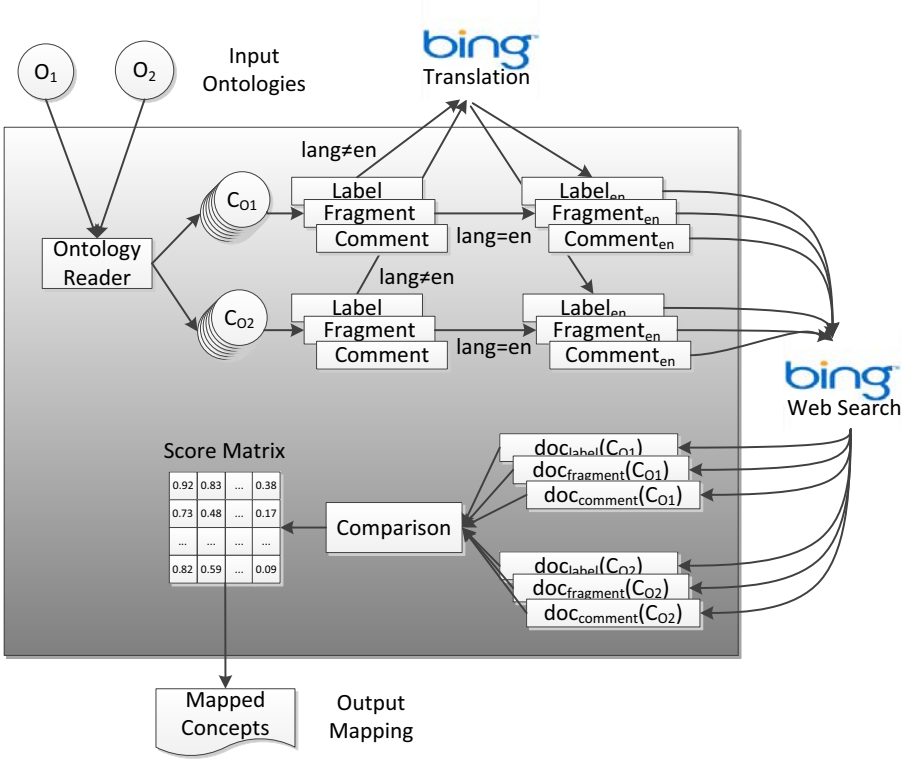


Fig. 2. Illustration of the *WeSeE-Match* matching process. Labels, fragments, and comments are extracted from the input ontologies, translated to English if necessary, and the documents are generated for each concept. A scoring matrix stores the maximum similarities for each pair of concepts. From that matrix, the final mapping is derived.

1.3 Adaptations made for the evaluation

No special adaptations have been made for OAEI 2012. The parameter τ was set to 0.55 for multi-lingual and to 0.6 for mono-lingual matching problems.

1.4 Link to the system and parameters file

The WeSeE-Match tool can be downloaded from `http://www.ke.tu-darmstadt.de/resources/ontology-matching/wesee-match`.

2 Results

2.1 Benchmark

The results on the benchmark set are those expected given the matcher's characteristics. Since the matcher is fully element-based, structural modifications of the ontolo-

gies (e.g., removing subclass relations) do not change the results. Furthermore, *WeSeE-Match* relies on natural language identifiers, labels, and comments. Removing those identifiers or replacing them by arbitrary strings creates ontologies where *WeSeE-Match* cannot identify meaningful alignments.

2.2 Anatomy

The results on the anatomy dataset show how background knowledge on the web helps identifying non-trivial mappings. For example, two concepts with the labels *anterior surface of the lens* and *lens anterior epithelium* are matched, one using an English, one a Latin name, as well as two concepts with labels *external ear* and *outer ear*, which are synonyms. As those names are likely to appear on similar web pages, *WeSeE-Match* is capable of identifying them as valid mappings.

2.3 Conference

The results on the conference track show how synonyms (like *Conference Attendee* and *Participant*, or *is reviewing* and *reviewer of paper*) are found by *WeSeE-Match*. The same mechanism, however, sometimes produces false positives of close terms like *Reviewer* and *Member PC*, since those often occur on similar web pages (i.e., conference websites and researchers' CVs).

A general observation is that the performance of *WeSeE-Match* is better with respect to classes than with respect to properties. This can be explained that class labels (such as *author*) make for more concise search terms than relations (such as *written by*).

2.4 Multifarm

Multi-lingual ontologies are well processed by *WeSeE-Match*, resulting in an average F-Measure of 0.41 across all language pairs. The worst results are achieved for Chinese-German (0.24), the best for English-French (0.56), where the latter is close to the performance of *WeSeE-Match* on the mono-lingual conference dataset. As discussed above, *WeSeE-Match* is well capable of identifying mappings between labels that are synonyms. It turns out that the Bing translation service used in *WeSeE-Match* does not provide exact translations, but merely closely related synonyms, such as *camera-ready version of the paper* and *final manuscript*, which are very problematic for string-based processing techniques. As discussed above, *WeSeE-Match* is particularly well suited for matching synonyms. Thus, the combination of translations (which may result in closely related terms) and matching via a web search engine is a good fit.

2.5 Library

Despite its general long run-time (see below), *WeSeE-Match* was capable of completing the larger library track. This track provides many different labels in three languages for most of the concepts, which leads to a lot of search engine requests, but the tool is capable of providing reasonable results at around the same level of quality as on the conference track. This shows that the larger number of labels available in the library track neither helps nor distracts *WeSeE-Match*.

2.6 Large Biomedical Ontologies

Due to a programming error, *WeSeE-Match* was not capable of completing that track.

3 General Comments

3.1 Comments on the Results

The results show that *WeSeE-Match* is capable of producing results that are competitive with state of the art matching tools, despite the very simple approach. Leveraging the knowledge of the world wide web for ontology matching thus appears to be a promising technique. The combination of machine translation and web search appears to be a good fit, because near-exact translations and synonyms are well matched by a search engine based approach.

Being one of the slowest matchers in OAEI, the downside of *WeSeE-Match* clearly is its runtime. However, it is important to notice that *WeSeE-Match* scales *linearly* with respect to runtime. In contrast to approaches such as Normalized Google Distance [1], which require a *quadratic* number of search engine invocations (to compute the number of pages on which a pair of concepts appears together), *WeSeE-Match* creates at most three search engine requests per concepts (one each for the label, the comment, and the URI fragment).

3.2 Possible Improvements of the System

At the moment, *WeSeE-Match* does not make any use of the input ontologies' structure, but is implemented as a purely element-based approach. Possible improvements would include the use of subclass relations as well as domains and ranges of properties. These could, e.g., be included as additional search terms. This could help improving the tool's performance on relations.

Although the tool has only one relevant parameter (the threshold τ), observations have shown that a good choice of this parameter strongly varies among the individual problems. Thus, the choices of this parameter for OAEI 2012 are compromises that provide reasonable, yet not optimal results for all problems. Automatic parameterization techniques [3] could help here in further improving the system's results.

4 Conclusion

The results of *WeSeE-Match* in the OAEI 2012 competition show that an algorithm based on a simple idea – using a standard web search engine and translation service – yields results that can keep up with competitive with tools that have much more complex underlying algorithms.

Given the long run-times, the approach is only applicable in scenarios that do not require real-time results. Furthermore, it is a possible candidate algorithm for dealing with hard-to-solve cases, where the simple cases are solved by faster algorithms. It is rather a candidate to be used in a tool with many matching algorithms to inspect those cases which cannot be handled by simpler algorithms.

References

1. Cilibrasi, R.L., Vitanyi, P.M.: The google similarity distance. IEEE Transactions on Knowledge and Data Engineering **19** (2007) 370–383
2. Paul, M., Yamamoto, H., Sumita, E., Nakamura, S.: On the importance of pivot language selection for statistical machine translation. In: 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers. (2009) 221–224
3. Ritze, D., Paulheim, H.: Towards an automatic parameterization of ontology matching tools based on example mappings. In: Proceedings of the 6th International Workshop on Ontology Matching, Bonn, Germany, October 24, 2011. Volume 814 of CEUR WS. (2011)

WikiMatch Results for OEAI 2012

Sven Hertling and Heiko Paulheim

Technische Universität Darmstadt
{hertling,paulheim}@ke.tu-darmstadt.de

Abstract. WikiMatch is a matching tool which makes use of Wikipedia as an external knowledge resource. The overall idea is to search Wikipedia for a given concept and retrieve all pages describing the term. If there is a large amount of common pages for two terms, then the concepts will have similar semantics. We make also use of the inter-language links between Wikipedias in different languages to match multilingual ontologies. The results show that this simple idea can keep up with state of the art tools. Moreover, the results on the Multifarm track depend on the Wikipedia's number of articles as well as the link amount to the Wikipedia of the other natural language to match. The growth of Wikipedia will thus help this matcher to improve the matching quality.

1 Presentation of the system

1.1 State, purpose, general statement

WikiMatch is an element-level ontology matching tool. It uses Wikipedia as a huge background knowledge to find out, how similar two concepts are. The algorithm extracts all labels, comments, and URI fragments, and uses Wikipedia's search function to retrieve an set of articles related to that term. If the intersection between such two sets is high, then we assume that the terms have something in common and are related to each other.

To also deal with multilingual ontologies, all language links of the returned articles are requested as a second step. For each language, the Jaccard coefficient of the two sets of articles retrieved is computed, as equation (1) shows.

$$sim(t_1, t_2) := max_{t_i \in \{label(c_i), fragment(c_i), comment(c_i)\}, i \in \{1,2\}} \frac{\#(S(t_1) \cap S(t_2))}{\#(S(t_1) \cup S(t_2))} \tag{1}$$

For the terms $t1$ and $t2$ the resulting article set S is computed. The maximum over all labels, comments and URI fragments are then the similarity measure for these terms.

If Wikipedia returns an suggestion for the term, a new query is made with this new search term. This is typically the case when entering a misspelled term in the search. An overview of the entire system is shown in Fig. 1.

1.2 Specific techniques used

Our first test was to search for the whole term in Wikipedia. We call this approach *simple search*. As a result the precision is high in contrast to the recall which is very low. To

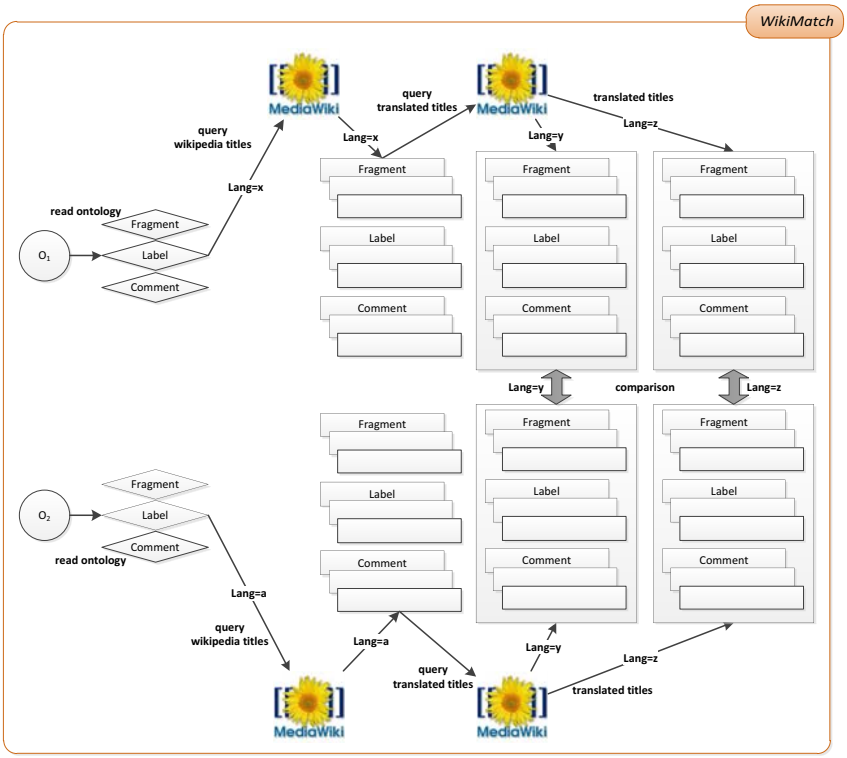


Fig. 1. Illustration of the matching process (see [1]). As a first all Wikipedia articles are requested for the language of the term. As a second step all language links from these articles are queried. The comparison of all these sets is per language. The maximum of the cross product of fragment, comment and label is returned.

improve the recall measure we have tried another search approach, i.e., splitting each term into individual tokens and searching for those tokens individually. For example, the query for the string *Passive_conference_participant* will therefore contain three single searches with $passive$, $conference$ and $participant$. Both search approaches are shown in Fig. 2 in pseudo code.

Our own tests showed that the *individual tokens search* (ITS) will result in a better recall, but a lower precision. To have a look at the F-Measure between the two approaches, the first idea of *simple search* can produce better values. Therefore this approach is was submitted.

1.3 Adaptations made for the evaluation

No adaptions for the evaluation are made.

```
float getsimilarity(term1, term2) {
  titlesForTerm1 = getAllTitles(term1);
  titlesForTerm2 = getAllTitles(term2);

  commonTitles = intersectionOf(titlesForTerm1, titlesForTerm2);
  allTitles = unionOf(titlesForTerm1, titlesForTerm2);

  return #(commonTitles) / #(allTitles);
}

List<WikipediaPage> getAllTitles(searchTerm) {
  removeStopwords(searchTerm);
  removePunctuation(searchTerm);

  if(simpleSearch) {
    resultList = searchWikipedia(searchTerm);
  }

  if(individualTokenSearch) {
    tokens = tokenize(searchTerm);
    for each token in tokens
      resultList = resultList + searchWikipedia(searchTerm);
  }

  for each page in results
    resultList = resultList + getLanguageLinks(page);

  return resultList;
}
```

Fig. 2. Pseudo code of *simple search* and *individual token search* (see [1]).

1.4 Link to the system and parameters file

The WikiMatch tool can be downloaded from `http://www.ke.tu-darmstadt.de/resources/ontology-matching/wikimatch`.

2 Results

2.1 Benchmark

Since our approach is entirely element-based, removing or replacing labels or comments results in lower F-Mesaure. By removing only one of the describing elements, WikiMatch deals also with the remaining literals and can provide good results. If there are neither labels nor comments, then this approach does not work. On the other hand, removing structural features, such as subclass relations, does not influence the results of WikiMatch.

2.2 Anatomy

The comparison with *StringEquiv* of the OAEI 2011.5 is not that well, because the recall is not much higher, but therefore the precision is very low (0.997 to 0.864). A nontrivial mapping that is found by our tool is *ophthalmic artery* and *Opthalmic_Artery*.

2.3 Conference

In the conference track, WikiMatch reached 0.6 F-Measure for ra1. This is better than the baseline2 from OAEI 2011.5. The same applies for ra2. Unfortunately, the conference domain is not well covered in Wikipedia to match special terms like *Chair_PC* and *ProgramCommitteeChair*. But through the suggestion feature it is possible to find a mapping between *Sponsorship* and *Sponzorship*.

2.4 Multifarm

On the Multifarm track, WikiMatch exploits the inter-language links from each returned article. Therefore a mapping between different languages can be found. The best results are achieved for matching English to Spanish (F-Measure 0.29), the worst for Chinese-German and Chinese-Portuguese (F-Measure 0.1).

The results on the Multifarm track strongly depend on the involved Wikipedia's sizes, in particular the number of articles and links to other Wikipedias. Fig. 3 depicts the results of WikiMatch in relation to the corresponding Wikipedias' article counts; Fig. 4 the results in relation to the number of links from the corresponding Wikipedias to other Wikipedias[1]. It can be observed that the results get better with larger and more strongly inter-linked Wikipedias.

As the number of articles and inter-Wikipedia links grow by around 2% per month (even more rapidly for Chinese, which is currently the smallest and least interlinked Wikipedia used in Multifarm), we expect the results of WikiMatch to improve just by the growth of Wikipedia. The trend lines in Fig. 3 and 4 indicate that about 500,000 additional articles and Wikipedia links lead to an increase of five percentage points in F-Measure. At the current growth rate of Wikipedia, this takes a little less than two years.

2.5 Library

The library track unfortunately did not finish within one week. The reason can be the calculation of the cross product between the concepts of the ontologies, or the generally long times required for looking up concepts in Wikipedia. This requires an more detailed look.

2.6 Large Biomedical Ontologies

Like the library track, the ontologies in this track are also too large handle by WikiMatch in its current version.

[1] Using numbers obtained from `http://stats.wikimedia.org/`

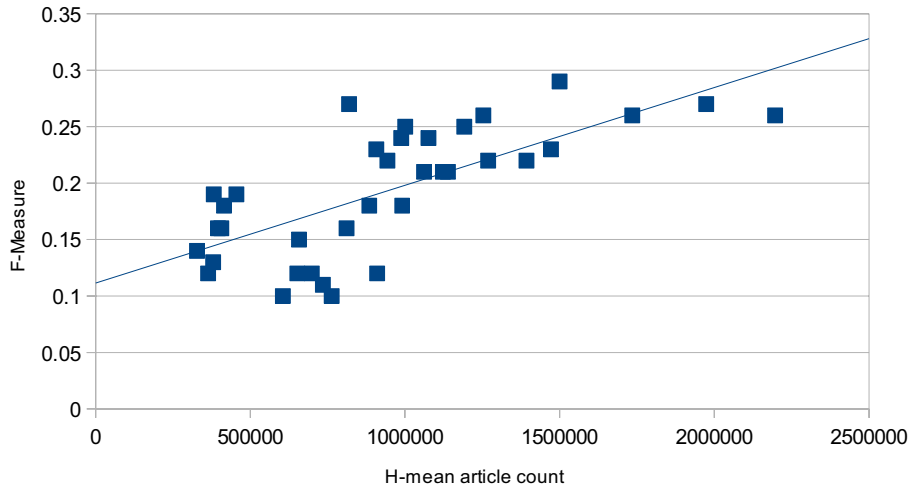


Fig. 3. Multifarm results in relation to the corresponding Wikipedias' article counts

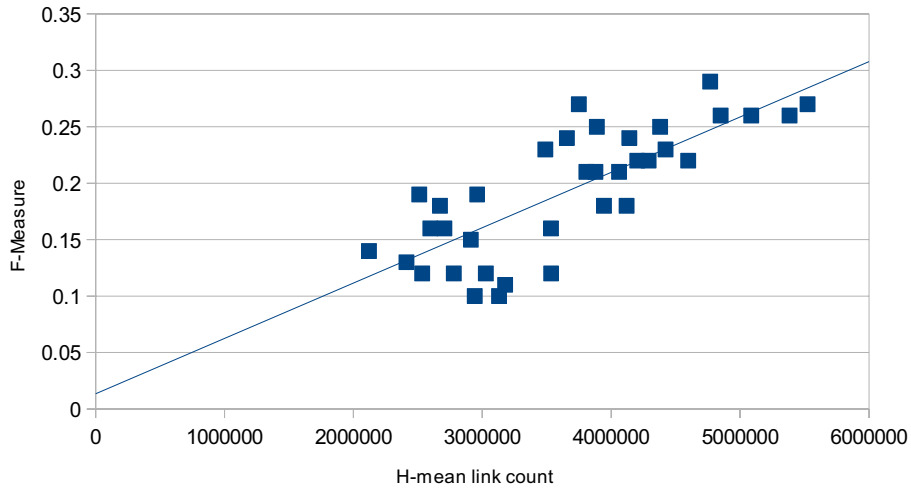


Fig. 4. Multifarm results in relation to the corresponding Wikipedias' inter-wiki link counts

3 General comments

3.1 Comments on the results

On Multifarm and conference track WikiMatch shows that a simple element based approach can keep up with state of the art tools. Especially using the inter-language links in Wikipedia looks like a promising approach to deal with multi-lingual ontologies. On large tracks the current approach does not scale well and did not finish in time.

In general, like most approaches using web data by querying the web at run-time, WikiMatch is rather slow compared to matchers working entirely internally or only use local resources.

3.2 Discussions on the way to improve the proposed system

For improving the approach, we envision to set threshold values dynamically, based on the matched ontologies. In order to cope with the run-time restrictions, it is possible to not use WikiMatch as a single matching approach, but to first match the easy cases (i.e., same or very similar terms) with string-level methods.

At the moment, WikiMatch only uses the page identifiers returned by the search, ignoring the text snippets, i.e., the portions of the Wikipedia pages that are relevant for the search term. Using those snippets, e.g., like WeSeE-Match does [2], could help leveraging the potential of WikiMatch more effectively.

4 Conclusion

With our work on WikiMatch, we have shown how a large general-purpose resource like Wikipedia can be used for ontology matching. Especially the cross-linking of different language Wikipedias is useful for multi-lingual ontology matching. Furthermore, we have seen that the results of WikiMatch improve with a growing size of Wikipedia – which in turn indicates that the results of WikiMatch will improve in the future merely by the growth of Wikipedia.

References

1. Hertling, S., Paulheim, H.: Wikimatch - using wikipedia for ontology matching. In: Seventh International Workshop on Ontology Matching (OM 2012). (2012)
2. Paulheim, H.: Wesee-match results for oeai 2012. In: Seventh International Workshop on Ontology Matching (OM 2012). (2012)

YAM++ – Results for OAEI 2012*

DuyHoa Ngo, Zohra Bellahsene

University Montpellier 2, INRIA, LIRMM
{duyhoa.ngo, bella}@lirmm.fr

Abstract. The YAM++ system is a self configuration, flexible and extensible ontology matching system. YAM++ takes advantages of many techniques coming from different fields such as machine learning, information retrieval, graph matching, etc. in order to enhance the matching quality. In this paper, we briefly present the YAM++ approach and its results on OAEI 2012 campaign.

1 Presentation of the system

YAM++ - (not) Yet Another Matcher is an automatic, flexible and self-configuring ontology matching system for discovering semantic correspondences between entities (i.e., classes, object properties and data properties) of ontologies. In YAM++ approach, multiple working strategies and matching techniques coming from machine learning, information retrieval, graph matching have been implemented in order to deal with both terminological and conceptual heterogeneity of ontologies. In the past, YAM++ achieved good results and gained high ranking positions in comparison with other participants in Benchmark, Conference and Multifarm tracks in OAEI 2011 and OAEI 2011.5 campaigns. This year, YAM++ participates in **six tracks** including **Benchmark**, **Conference**, **Multifarm**, **Library**, **Anatomy** and **Large Biomedical Ontologies** tracks.

1.1 State, purpose, general statement

The major principle of the matching strategy in YAM++ approach is utilizing as much useful information as possible of entities in ontologies effectively and efficiently. In the previous YAM++ (OAEI **2011** version), it is a combination of machine learning and graph matching techniques. In particular, Decision Tree learning model is used to combine different terminological similarity measures, whereas, a similarity propagation method is used to discover mappings by exploiting structural information of entities. The drawback of the previous version of YAM++ lies in its low performance in terms of time and high memory consuming. Therefore, it was inapplicable for large scale ontology matching scenarios.

In the current version (OAEI **2012**), several changes of YAM++ have been done. Firstly, since OAEI 2011.5 campaign, we have proposed new similarity measures based on techniques coming from information retrieval field in order to compare short and long texts. These measures are an alternative solution to the machine learning method,

* Supported by ANR DataRing ANR-08-VERSO-007-04.

which was used in the YAM++ 2011 version, in the case where no training data is available. Next, a semantic verification component have been added in YAM++ in order to enhance the matching quality. Finally, a candidate filtering component have been designed for reducing computational space when dealing with large scale ontology matching scenarios.

1.2 Specific techniques used

In this section, we will briefly describe the workflow of YAM++ and its main components, which are shown in Fig.1.

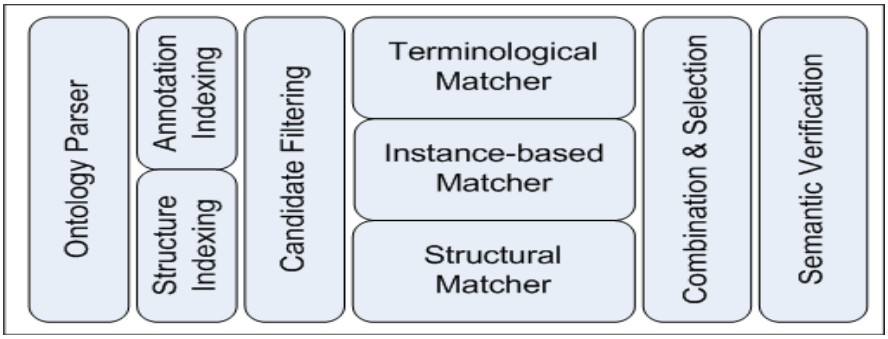


Fig. 1. Main components of YAM++ system

In YAM++ approach, a generic workflow for a given ontology matching scenario is as follows.

1. Input ontologies are loaded and parsed by a **Ontology Parser** component;
2. Information of entities in ontologies are indexed by the **Annotation Indexing** and the **Structure Indexing** components;
3. **Candidates Filtering** component filters out all possible pairs of entities from the input ontologies, whose descriptions are highly similar;
4. Among those candidate mappings, the **Terminological Matcher** component produces a set of mappings by comparing the annotations of entities;
5. The **Instance-based Matcher** component supplements new mappings through shared instances between ontologies;
6. In YAM++, matching results of the **Terminological Matcher** and the **Instance-based Matcher** are aggregated into an element level matching result. The **Structural Matcher** component then enhances element level matching result by exploiting structural information of entities;
7. The mapping results obtained from the three matchers above are then combined and selected by the **Combination & Selection** component to have a unique set of mappings;
8. Finally, the **Semantic Verification** component refines those mappings in order to eliminate the inconsistent ones.

Ontology Parser To read and parse input ontologies, YAM++ uses OWLAPI open source library. In addition, YAM++ makes use of Pellet - an OWL 2 Reasoner in order to discover hidden relations between entities in ontologies. Here, the whole ontology is stored in the main memory.

Annotation Indexing In this component, all annotations information of entities such as ID, labels and comments are extracted. The languages used for representing annotations are considered. In the case where input ontologies use different languages to describe the annotations of entities, a multilingual translator (**Microsoft Bing**) is used to translate those annotations to English. Those annotations are then normalized by tokenizing into set of tokens, removing stop words, and stemming. Next, tokens are indexed in a table for future use.

Structure Indexing In this component, the main structure information such as IS-A and PART-OF hierarchies of ontologies are stored. In particular, YAM++ assigns a compressed bitset values for every entity of the ontologies. Through the bitset values of each entity, YAM++ can fast and easily gets its ancestors, descendants, etc. A benefit of this method is to easily access to the structure information of ontolgy and minimize memory for storing it. After this step, the loaded ontologies can be released to save main memory.

Candidates Filtering The aim of this component is to reduce the computational space for a given scenario, especially for the large scale ontology matching tasks. In YAM++, two filters have been designed for the purpose of performing terminology-based matchers efficiently.

- A **Description Filter** is a search-based filter, which filters out candidate mappings before computing the real similarity values between the description of entities. Here, a description of an entity consists of its labels, synonym labels and comments. The idea of this filter is as follows. Firstly, the descriptions of all entities in the bigger size ontology are indexed by **Lucene** search engine. For each entity in the smaller size ontology, a multiple terms query created by tokens within the description of this entity is executed in order to find the **top-K** similar entities.
- A **Label Filter** is used to fast detect candidate mappings, where labels of entities in each candidate mapping are similar or differ in maximum two tokens. The intuition is that if two labels of two entities differ by more than three tokens, any string-based method will produce a low similarity score value. Then, these entities are highly unmatched.

Terminological Matcher In YAM++, the **Terminological Matcher** component is compounded by two sub-matcher namely **Label Matcher** and **Context Profile Matcher**.

- The **Label Matcher** splits all labels of entities into tokens and calculates the information content of each token in the whole ontology. Then, it makes use of Tversky similarity measure to compute similarity score between labels of entities. Let we

explain how this method works by an example with two entities: `cmt.owl#Co-author` and `conference.owl#Contribution_co-author`. After splitting and normalizing labels, we have 2 sets of tokens such as: {`coauthor`} and {`coauthor`, `contribution`}. Token `coauthor` appears in each input ontology only one time, whereas, token `contribution` appears 10 times among 60 concepts in the ontology `conference.owl`. Therefore, the information content of the token `contribution` is less than that of the token `coauthor`. In particular, the normalized TFIDF weights of each token inside the input ontologies are equal: $\{w_{coauthor} = 1.0\}$, $\{w_{coauthor} = 1.0, w_{contribution} = 0.34\}$. Two sets of tokens share only token `coauthor`, then the similarity computed by Tversky method is $\frac{1.0+1.0}{1.0+1.0+0.34} = 0.855$.

- The **Context Profile Matcher** is used to compute similarity value of entities by comparing their context profiles, which are normally a long text. Like in the first YAM++ version, a context profile of an entity may be an **Individual Profile**, **Semantic Profile** or **External Profile**. We refer to [4] for more detail about the construction of these profiles and computation of the similarity between them.

Instance-based Matcher In many situation, the input ontologies provide data (instances), therefore, the aim of the **Instance-based Matcher** is to discover new mappings which are complement to the result obtained from the **Terminological Matcher**. Basically, the **Instance-based Matcher** is not changed in from the first YAM++ version to the current version. Therefore, for saving space, we refer to section **Extensional Matcher** of [3] for more detail.

Structural Matcher The **Structural Matcher** component contains two similarity propagation methods namely Similarity Propagation and Confidence Propagation.

- **Similarity Propagation** method is a graph matching method, which inherits the main features of the well-known **Similarity Flooding** algorithm [2]. The only difference is about transforming an ontology to a directed labeled graph. This matcher is not changed from the first YAM++ version to the current version. Therefore, for saving space, we refer to section **Similarity Flooding** of [3] for more detail.
- The intuition of the **Confidence Propagation** method is as follows. Assume $\langle a_1, b_1, \equiv, c_1 \rangle$ and $\langle a_2, b_2, \equiv, c_2 \rangle$ are two initial mappings, which are maybe discovered by the element level matcher (i.e., terminological matcher or instance-based matcher). If a_1 and b_1 are ancestors of a_2 and b_2 respectively, then after running confidence propagation, we have $\langle a_1, b_1, \equiv, c_1 + c_2 \rangle$ and $\langle a_2, b_2, \equiv, c_2 + c_1 \rangle$. Note that, confidence values are propagated only among collection of initial mappings.

In YAM++, the aim of the **Similarity Propagation** method is discovering new mappings by exploiting as much as possible the structural information of entities. This method is used for a small scale ontology matching task, where the total number of entities in each ontology is smaller than 1000. In contrary, the **Confidence Propagation** method supports a **Semantic Verification** component to eliminate inconsistent mappings. This method is mainly used in a large scale ontology matching scenario.

Mappings Combination and Selection The aim of the **Mappings Combination and Selection** component is to produce a unique set of mappings from matching results obtained by terminological matcher, instance-based matcher and structural matcher. In this component, a **Dynamic Weighted Aggregation** method have been implemented. Given an ontology matching scenario, it automatically computes a weight value for each matcher and establishes a threshold value for selecting the best candidate mappings. The main idea of this method can be seen in [3] for more detail.

Semantic Verification After running the similarity or confidence propagation on overall candidate mappings, the final achieved similarity values reach a certain stability. Based on those values, YAM++ is able to remove inconsistent mappings with more certainty. There are two main steps in the **Semantic Verification** component such as (i) identifying inconsistent mappings, and (ii) elimination inconsistent mappings.

In order to identify inconsistencies, several semantic conflict patterns have been designed in YAM++ as follows:

- Two mappings $\langle a_1, b_1\rangle$ and $\langle a_2, b_2\rangle$ are crisscross conflict if a_1 is an ancestor of a_2 in ontology O_1 and b_2 is an ancestor of b_1 in ontology O_2.
- Two mappings $\langle a_1, b_1\rangle$ and $\langle a_2, b_2\rangle$ are disjointness subsumption conflict if a_1 is an ancestor of a_2 in ontology O_1 and b_2 disjoints with b_1 in ontology O_2.
- A property-property mapping $\langle p_1, p_2\rangle$ is inconsistent with respect to alignment A if $\{Doms(p_1) \times Doms(p_2)\} \cap A = \emptyset$ and $\{Rans(p_1) \times Rans(p_2)\} \cap A = \emptyset$ then (p_1, p_2), where $Doms(p)$ and $Rans(p)$ return a set of domains and ranges of property p.
- Two mappings $\langle a, b_1\rangle$ and $\langle a, b_2\rangle$ are duplicated conflict if the cardinality matching is 1:1 (for a small scale ontology matching scenario) or the semantic similarity $SemSim(b_1, b_2)$ is less than a threshold value θ (for a large scale matching with cardinality 1:m).

In order to eliminate inconsistent mappings, a **Greedy Selection** method is used. The idea of this method is that it iteratively selects the mapping with the highest confidence value, which does not conflict with the mappings already selected before.

In YAM++, we used **Alcomo** [1] - an effective open source tool to eliminate inconsistent mappings for the first three conflict patterns. For the last pattern, a supplementary method called **Duplicate Removing** have been implemented. In this method, semantic similarity of two classes in ontology is computed by Resnik method [5], where an information content value of a class is computed by an intrinsic method described in [6].

1.3 Adaptations made for the evaluation

Before running the matching process, YAM++ analyzes the input ontologies and adapts itself to the matching task. In particular, if annotations of entities in input ontologies are described by different languages, YAM++ automatically translates them in English. If the number of entities in input ontologies is smaller than 1000, YAM++ is switched to small scale matching regime, otherwise, it runs with large scale matching regime. The main difference between two regime lies in the **Structural Matcher** and **Semantic Verification** components as we discussed above.

1.4 Link to the system and parameters file

A SEALS client wrapper for YAM++ system and parameter files can be download at: http://www2.lirmm.fr/d̃ngo/YAMplusplus2012.zip. See the instructions in tutorial from SEALS platform[1] to test our system.

1.5 Link to the set of provided alignments (in align format)

The results of all tracks can be downloaded at: http://www2.lirmm.fr/d̃ngo/ YAMplusplus2012Results.zip.

2 Results

In this section, we present the evaluation results obtained by running YAM++ with SEALS client with **Benchmark**, **Conference**, **Multifarm**, **Library**, **Anatomy** and **Large Biomedical Ontologies** tracks. All experiments are executed by YAM++ with SEALS client version 4.1 beta and JDK 1.6 on PC Intel 3.0 Pentium, 3Gb RAM, Window XP SP3.

2.1 Benchmark

In OAEI 2012, Benchmark includes 2 open tests (i.e. biblio, finace) and 3 blind tests (i.e., Benchmark 2, 3, 4). Table 1 shows the results of YAM++ running on the Benchmark data set.

Test set	H-mean Precision	H-mean Recall	H-mean Fmeasure
Biblio	0.98	0.72	0.83
Benchmark 2	0.96	0.82	0.89
Benchmark 3	0.97	0.76	0.85
Benchmark 4	0.96	0.72	0.83
Finance	0.97	0.84	0.90

Table 1. YAM++ results on pre-test Benchmark track

2.2 Conference

Conference track now contains 16 ontologies from the same domain (conference organization) and each ontology must be matched against every other ontology. This track is an open+blind, so in the Table 2, we can only report our results with respect to the available reference alignments

Test set	H-mean Precision	H-mean Recall	H-mean Fmeasure
Conference	0.802	0.692	0.743

Table 2. YAM++ results on Conference track

[1] http://oaei.ontologymatching.org/2012/seals-eval.html

2.3 MultiFarm

The goal of the MultiFarm track is to evaluate the ability of matcher systems to deal with multilingual ontologies. It is based on the OntoFarm dataset, where annotations of entities are represented in different languages such as: English (en), Chinese (cn), Czech (cz), Dutch (nl), French (fr), German (de), Portuguese (pt), Russian (ru) and Spanish (es). For saving space, we do not list all results here. Instead, the results of YAM++ can be found at at SEALS result repository[2].

2.4 Anatomy

The Anatomy track consists of finding an alignment between the Adult Mouse Anatomy (2744 classes) and a part of the NCI Thesaurus (3304 classes) describing the human anatomy. Table 3 shows the evaluation result and runtime of YAM++ on this track.

Test set	Precision	Recall	Fmeasure	Run times
Anatomy	0.944	0.868	0.904	201 (s)

Table 3. YAM++ results on Anatomy track

2.5 Library

The library track is a real-word task to match the STW (6575 classes) and the TheSoz (8376 classes) thesaurus. Table 4 shows the evaluation result and runtime of YAM++ against an existing reference alignment on this track.

Test set	Precision	Recall	Fmeasure	Run times
Library	0.595	0.750	0.663	759 (s)

Table 4. YAM++ results on Library track

2.6 Large Biomedical Ontologies

This track consists of finding alignments between the Foundational Model of Anatomy (FMA), SNOMED CT, and the National Cancer Institute Thesaurus (NCI). There are 9 sub tasks with different size of input ontologies, i.e., small fragment, large fragment and the whole ontologies. Table 5 shows the evaluation results and run times of YAM++ on those sub tasks.

3 General comments

This is the third time YAM++ participates to the OAEI campaign. We found that SEALS platform is a very valuable tool to compare the performance of our system with the others. Besides, we also found that OAEI tracks covers a wide range of heterogeneity in ontology matching task. They are very useful to help developers/researchers to develop their semantic matching system.

[2] http://www.seals-project.eu/

Test set	Precision	Recall	Fmeasure	Run times
Small FMA - NCI	0.980	0.848	0.9093	482 (s)
Large FMA - NCI	0.923	0.821	0.869	1908 (s)
Whole FMA - NCI	0.906	0.821	0.861	3864 (s)
Small FMA - SNOMED	0.972	0.693	0.809	1990 (s)
Large FMA - SNOMED	0.879	0.684	0.769	7709 (s)
Whole FMA - SNOMED	0.878	0.683	0.768	9907 (s)
Small SNOMED - NCI	0.951	0.604	0.739	5643 (s)
Large SNOMED - NCI	0.864	0.599	0.708	13233 (s)
Whole SNOMED - NCI	0.859	0.599	0.706	17690 (s)

Table 5. YAM++ results on Large Biomedical Ontologies track

3.1 Comments on the results

The current version of YAM++ has shown a significant improvement in terms of matching quality and runtime with respect to the previous version. In particular, the H-mean Fmeasure value of the Conference track increases $0.74 - 0.65 = 0.09$; this version is able to run with not only scalability dataset but also very large scale dataset (i.e., Library, Biomedical ontologies).

4 Conclusion

In this paper, we have presented our ontology matching system called YAM++ and its evaluation results on different tracks on OAEI 2012 campaign. The experimental results are promising and show that YAM++ is able to work effectively and efficiently with real-world ontology matching tasks. In near future, we continue improving the matching quality and efficiency of YAM++. Furthermore, we plan to deal with instance matching track also.

References

1. Christian Meilicke. Alignment incoherence in ontology matching phd. thesis. In *University of Mannheim, Chair of Artificial Intelligence*, 2011.
2. Sergey Melnik, Hector Garcia-Molina, and Erhard Rahm. Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In *ICDE*, pages 117–128, 2002.
3. DuyHoa Ngo, Zohra Bellahsene, and Remi Coletta. Yam++ results for oaei 2011. In *OM*, 2011.
4. DuyHoa Ngo, Zohra Bellasene, and Remi Coletta. A generic approach for combining linguistic and context profile metrics in ontology matching. In *ODBASE Conference*, 2011.
5. Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *IJCAI*, pages 448–453, 1995.
6. David Sánchez, Montserrat Batet, Aida Valls, and Karina Gibert. Ontology-driven web-based semantic similarity. *J. Intell. Inf. Syst.*, pages 383–413, 2010.

A modest proposal for data interlinking evaluation*

Jérôme Euzenat

INRIA & LIG (Jerome.Euzenat@inria.fr)

Data interlinking, i.e., finding links between different linked data sets, is similar to ontology matching in various respects and would benefit from widely accepted evaluations such as the Ontology Alignment Evaluation Initiative. Instance matching evaluation has been performed as part of OAEI since 2009. Yet, there has been so far few participants to IM@OAEI and there is still no largely acknowledged benchmark for data interlinking activites.

In order to secure more participation, we analyse the specificities of data interlinking and propose diverse modalities for evaluating data interlinking.

1 The problem

The data interlinking problem could be described in the same way as the ontology matching problem was:

Given two linked data sets, usually tied to their vocabularies (or ontologies),
Generate a link set, i.e., a set of sameAs links between entities of the two data sets.

We concentrate on sameAs links because these are by far the most important links to be retrieved when interlinking.

In terms of evaluation, the same kind of procedure as in ontology matching may be used by comparing the provided link set to a reference link set. Measures such as precision, recall or time and memory consumption may be used.

Given the size of the data, it is difficult to provide correct and more specifically complete reference link sets. The OAEI 2011 IM task solved the problem by using links already provided by data providers. This is valuable when these links are curated manually. However, the quality of these links may still be questioned.

2 Specific interlinking features

We present some features of the way data interlinking is performed nowadays, that distinguish it from ontology matching.

Blocking vs. matching Ontology matching, when confronted to large ontologies, cannot start upfront comparing instances with a similarity measure. The same applies to data interlinking. Hence many data interlinking procedures are designed in two steps:

blocking divides the sets of pairs of resources into subsets called blocks in which matching resources should be part;

*This work has been partially supported by the French National Research Agency (ANR) under grant ANR-10-CORD-009 (Datalift). A longer version of the proposal is available at `ftp://ftp.inrialpes.fr/pub/exmo/publications/euzenat2012b.pdf`

matching compares entities in the same block in order to decide if they are the same.

Since these two steps are clearly different, well identified, and it is possible to use different matching methods with different blocking techniques, it is useful to evaluate independently the capacities of these two techniques.

In particular, from a reference link set, it is possible to determine how many pairs in the link set a particular blocking technique misses (blocking recall) and how many non necessary pairs a blocking technique imposes to compare (blocking precision). Similarly, a matching technique could be evaluated with a given block structure if this is necessary: the evaluation can be achieved by comparing the part of the reference alignments which can be found from the given blocks.

Scalability Linked data has to deal with large amounts of data. Even if this is also the case in ontology matching, automatic data interlinking is really useful with large amounts of data. So, besides qualitative evaluation, it is critical to assess the behaviour of interlinking tools when data sizes get larger.

Learning Another effect of the size of linked data is that learning is more relevant, mainly for two reasons:

- The size of the data makes it difficult to study it for choosing the best approach and after extracting a training sample, much work remains to be done;
- The regularity of the data facilitates machine learning efficiency.

So, it is not surprising that learning methods are successful in data interlinking. This provides incentive to evaluate data interlinking techniques using machine learning. For that purpose, it is necessary to provide tests in which a part of the reference link set is provided as training set to the systems which have then to deliver a complete link set.

Instance and ontology matching Data interlinking is dependent on the vocabularies used in data sets. This vocabulary may be described by a schema or an ontology or not described explicitly. Some matchers may be specialised for some of these situations and it may be useful to recognise this by providing different evaluation tasks. In particular, it seems useful to test these configurations:

- without published vocabulary (or ontology),
- with the same vocabulary or, alternatively, with aligned vocabularies,
- with different vocabularies, without alignment (as in the IIMB data set of IM@OAEI).

3 Conclusions

In conclusion, it seems that in addition to or in combination with existing tasks provided in IM@OAEI, such benchmarks should consider including:

- scalability tests (retaining one tenth, one hundredth, one thousandth of the data);
- training sets for tools using machine learning;
- separating the evaluation of blocking and matching for users who specifically consider one of these aspects only;
- tests with no ontology, the same ontology and different ontologies.

A Comparison of Complex Correspondence Detection Techniques

Brian Walshe, Rob Brennan, Declan O'Sullivan

FAME & Knowledge and Data Engineering Group,
School of computer Science and Statistics,
Trintiy College Dublin.

{walshebr|rob.brennan|declan.osullivan}@scss.tcd.ie

Abstract. One to one correspondences between entities are not always sufficient to describe the true relationship between related entities in diverse ontologies, and complex correspondences are needed instead. We demonstrate the types of complex correspondence occurring between two LOD sources and compare techniques for discovering these complex correspondences.

1 Motivation and Background

Most alignment research focuses on one-to-one correspondences between named ontology elements [1], but these are not always sufficient for performing many integration tasks [2]. Data values, for example, may need some form of translation, or some form of condition may be required to scope a broader concept to correspond with a narrower one. These correspondences, which contain conditions or transformations, are known as *complex correspondences*.

There are many known patterns of complex correspondence [2]. Conditional correspondences – where instances of a concept in one ontology are related to a corresponding concept in the other ontology only if they have a particular value for a given attribute – include Class by Attribute Type (CAT), Class by Attribute Value (CAV), and Class by Attribute Existence (CAE). Similarly, Class by Attribute Path Correspondences (PATH) occur when some path of attributes must be followed before the scope of the more general concept can be narrowed. Correspondences where the value of an attribute must be altered in some way are called Attribute Transformation Correspondences (ATC).

In a sample of 50 concepts from YAGO2 [3], six of these concepts corresponded to equivalent concepts in the DBpedia [4] ontology, and 14 concepts required a Class by Attribute Value correspondence. Twenty-one concepts from YAGO2 corresponded with DBpedia concepts with broader scope which could not be narrowed with a correspondence pattern. Six YAGO2 concepts were aligned with DBpedia instances. We found no cases of CAT or PATH correspondences.

Approach	CAV	CAT	CAE	ATC	PATH
Pattern Fitting	Boolean values	Yes	No	No	Yes
MRDM	Yes	Yes	Yes	No	Yes
Model Fitting	Yes	Yes	Yes	Numerical	No

Table 1. Types of correspondence patterns each approach can detect.

2 Detecting Complex Correspondences

Approaches to detecting complex correspondences include a pattern based approach [5], multi relational data mining (MRDM) [6] and our model based approach [7]. Each approach differs in the particular types of correspondence it can detect, and these differences are outlined in table 1. The pattern based approach is the least flexible. For attribute value based patterns it is only capable of detecting cases where attributes have Boolean values. Each of the complex correspondences we found between DBpedia and YAGO2 use non-Boolean attributes, and so it could not detect these. The MRDM approach is more flexible, and is theoretically capable of finding most correspondence patterns listed in section 1, except value transformation patterns. Only the model fitting approach is capable of detecting value transformation correspondences. The current implementation can detect numerical transformations, but the approach could be extended to also detect transformations such as string splitting.

Acknowledgement: This research is supported by the Science Foundation Ireland (Grant 08/SRC/I1403) as part of the FAME Strategic Research Cluster.

References

1. Shvaiko, P., Euzenat, J.: Ontology matching: state of the art and future challenges. IEEE Transactions on Knowledge and Data Engineering. preprint, (2012).
2. Scharffe, F., Fensel, D.: Correspondence patterns for ontology alignment. Knowledge Engineering: Practice and Patterns. 83–92 (2008).
3. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: A large ontology from wikipedia and wordnet. Web Semantics: Science, Services and Agents on the World Wide Web. 6, 203–217 (2008).
4. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia - A crystallization point for the Web of Data. Web Semantics: Science, Services and Agents on the World Wide Web. 154-165 (2009).
5. Ritze, D., Meilicke, C., Sváb-Zamazal, O., Stuckenschmidt, H.: A pattern-based ontology matching approach for detecting complex correspondences. Proc. of Int. Workshop on Ontology Matching (OM) (2009).
6. Qin, H., Dou, D., Lependu, P.: Discovering Executable Semantic Mappings Between Ontologies. 2007 OTM Confederated international conference on On the move to meaningful internet systems: CoopIS, DOA, ODBASE, GADA, and IS. pp. 832-849 (2007).
7. Walshe, B.: Identifying Complex Semantic Matches. 9th Extended Semantic Web Conference. pp. 849-853 (2012).

On Ambiguity and Query-Specific Ontology Mapping

Aibo Tian, Juan F. Sequeda, and Daniel P. Miranker

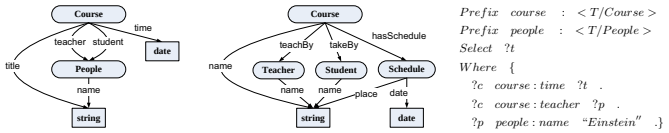
Department of Computer Science, The University of Texas at Austin
Austin, Texas, USA
atian@utexas.edu, {jsequeda, miranker}@cs.utexas.edu

Abstract. In the course of developing an ontology-based data integration system (OBDI) that includes automatic integration of data sources, and thus, includes algorithmic ontology mapping, we have made the following observations. A mapping method may determine that an entity in one ontology maps with equal likelihood to two or more entities in the other ontology. The mapping and reformulation of certain queries is correct only if one pairing is chosen. The correct choice may be different for different queries. Finally, the query itself may lend additional semantics that correctly resolve the ambiguity.
These observations suggest a targeted ontology mapping problem, *query-specific ontology mapping*. In addition to the two ontologies, a query serves as a third argument to the mapping algorithm. Further, the mapping algorithm need not produce a complete mapping, but only a partial mapping sufficient to correctly reformulate the query. We detail a number of open issues on how this problem statement might be refined, and consider features of its evaluation.

Ambiguity in Ontology Mapping: Consider the idealized representation (Fig. 1) of a critical issue in the automatic integration of new data sources in an OBDI system. T and S respectively represent target and data source ontologies. Looking at the ontologies alone, there is insufficient information to determine if the class T:People should be mapped to S:Teacher or to S:Student. A third possibility is a one-to-many mapping entailing both. Given the SPARQL query (Fig. 1c), it becomes clear that the query should be reformulated using *only* the mapping {T:People = S:Teacher}. A complementary query about students should be reformulated using *only* the complementary mapping. Thus, any static chose of one mapping will yield reformulated queries that return incorrect results.

Formulations of Query-Specific Ontology Mapping: In our system we compute a similarity matrix between all entities in the two ontologies [3]. The details may be borrowed from any ontology mapping algorithm that includes this step [2]. Given a query on the target ontology, our system uses a joint probability model to identify a maximal scoring, partial mapping that covers the target ontology entities mentioned in the query or that are needed to reformulate the query. Thus, our solution can be characterized as one that takes three arguments, and produces a partial mapping specific to the query.

There are at least two other approaches that may be considered and that produce a complete mapping and thus retain more of the standard definition of ontology matching. First is to consider complex mappings. For example, instead of choosing {T:People =



(a) Ontology T (b) Ontology S (c) SPARQL query

Fig. 1. Example ontologies and SPARQL query.

S:Teacher} or {T:People = S:Student}, the mapping system can detect "Teacher is the People who teaches" (similar for Student). However, to the best of our knowledge, there is no automatic system that can detect this kind of complex mapping.

Another approach may consider an entire workload of queries, as a batch or as a continual pay-as-you-go refinement. In other words, a complete mapping is determined, but the information in a set of queries is used to bias the choices made. As many applications comprise a set of dynamic web pages, their query set is easily identified. Consider the example and a course selection application. Since students are often interested in who is teaching a class, (and their grading policy), and privacy laws disallow revealing their fellow student's enrollment, the mapping {T:People = S:Teacher} would always be correct. Incremental, pay-as-you go, solutions could integrate crowd-sourcing.

The pedagogical example's brevity shouldn't be used to diminish the problem's importance. Comparing to Clio's[1] algorithms our system demonstrates favorable results [1, 3]. Inspection of individual results suggests that resolving ambiguity is the primary source of improvement, and can be significant. However measuring the quality of the solutions, as a whole, and quantifying the frequency of ambiguity poses its own set of problems. Gold standard baselines must include queries and correct mappings. OAEI benchmarks cannot be used directly. Correct query reformulation may not require a unique mapping. Entity level ambiguity may not manifest wrt query reformulation, making it hard to identify through manual curation. To date, we have created three such test cases[2]. The test suite accommodates the unique mapping problem by including additional partial mappings and including test data corresponding query results. Not all ambiguity may be revealed. Our inspection of individual results looked at the discrepancies between the two systems. False negatives are not quantifiable.

References

1. R. Fagin, L. Haas, M. Hernández, R. Miller, L. Popa, and Y. Velegrakis. Clio: Schema mapping creation and data exchange. *Conceptual Modeling: Foundations and Applications*, 2009.
2. P. Shvaiko and J. Euzenat. Ontology matching: state of the art and future challenges. *IEEE Transactions on Knowledge and Data Engineering*, 2012.
3. A. Tian, J. F. Sequeda, and D. P. Miranker. Query specific ontology matching. Technical report, Department of Computer Science, University of Texas, 2012.

[1] Clio is an automatic relational schema mapping system. However, the algorithms are applicable to ontologies.

[2] The test cases are available, see http://www.cs.utexas.edu/~atian/page/dataset.html

Utilizing Regular Expressions for Instance-Based Schema Matching

Benjamin Zapilko, Matthäus Zloch, and Johann Schaible

GESIS - Leibniz Institute for the Social Sciences
Unter Sachsenhausen 6-8, 50667 Cologne, Germany
{benjamin.zapilko,matthaeus.zloch,johann.schaible}@gesis.org

Abstract. Statistical data consists mostly of numerical values, entries of codelists like country codes or acronyms for gender. Such values are typically described according to specific patterns. In this paper we present a novel approach for instance-based schema matching, where regular expressions are utilized for matching patterns of instance values.

1 Motivation and Background

In various domains, e.g. the social sciences, the matching of statistical data is a typical task. Schema elements of statistical data, e.g. rows or columns of a spreadsheet, are named usually by simple and short labels, sometimes even with abbreviated terms. However, the structure and semantics of their instances (e.g. numerical values, entries of codelists) differ in various aspects from text-heavy data. Instances are often described by a specific syntactical pattern, e.g. dates consist of numerical values divided by periods or slashes or a three-letter code for a geographical area.

For instance-based schema matching [3] states that different domains reveal new challenges like treating new types of information resources, e.g. spatial or temporal information or domain-specific constrains. According to [2] especially domain-specific values, significant occurrences and patterns of values are relevant characteristics to be considered at instance level, as well as integrity constraints for schema elements and their instance values. In [1] the matching process is enhanced by applying a constraint-based matching. Moreover, regular expressions and catchwords are considered for instance-based schema matching in [4]. We focus on statistical data, where the potential of patterns and regular expressions for schema matching can be fully exposed.

2 Schema Matching using Regular Expressions

By utilizing pattern classes our approach considers two schema elements as a match, if their instances can be expressed via at least one regular expression of the same pattern class. We define multiple pattern classes, which correspond to a specific data element, e.g. dates, age groups or geographical codes,

and contain various patterns for describing this data element. For a data element "date" different patterns might be e.g. `[0-9]{4}`, `[0-9]{2}-[0-9]{4}` or `[0-9]{2}.[0-9]{2}.[0-9]{4}`. Each pattern is expressed as a regular expression and is assigned a weighting, which states the accuracy of the pattern to compass typical instances of the data element. Inside a pattern class the regular expressions are sorted by their weightings in descending order.

We assume two datasets M and N with their schema elements $S_M \in M$ and $S_N \in N$. The pattern classes C_x with $C_x = \{(regex, \omega) | regex \text{ matches } x, 0 < \omega < 1\}$ contain multiple regular expressions $regex$ describing the statistical data element x of the class. They are accompanied with a weighting ω.

For each pattern class C_x, we compute an average weighting for every schema element S_M and S_N. This average weighting indicates how often instances of the schema element can be expressed by a pattern of the class. Hereby, as soon as an instance can be expressed by a $(regex, \omega) \in C_x$, the value of ω is added to the sum of all weightings, whose regular expressions previously matched another instance from this same schema element, resulting in the final $\sum_0 \omega$. The average is then retrieved by normalizing this sum regarding the total number of instances inside this particular schema element. For each S_M, this is $avg(S_M) = \frac{\sum_0 \omega}{|\text{Instances in } S_M|}$. For S_N the average is calculated analogously. If this average weight is not 0, the schema element is collected among its average weight in a set. We define these sets as M_x and N_x with $M_x = \{(S_M, avg(S_M)\}$ and $N_x = \{(S_N, avg(S_N)\}$.

The Cartesian product of M_x and N_x is computed and added to $Matches_x$, in which a triple $(S_M, S_N, avg(S_M) * avg(S_N))$ defines a match between a S_M and a S_N with the probability of $avg(S_M) * avg(S_N)$. Finally, the result set $Matches_x$ contains all matches between two datasets M and N.

Our approach has been implemented in Java using the JENA API. The source code and an executable jar file are available at https://github.com/mazlo/smurf. In first experiments with real-world statistical data we obtained better results for matching schema elements than other existing matching systems. A detailed evaluation with generic test datasets is currently work-in-progress. We aim to extend our approach to extract patterns from instance values and to generate weightings automatically. Feature extraction from instance values can enhance our approach in computing weightings and in assigning regular expressions to adequate pattern classes.

References

1. Engmann, D.; Maßmann, S. Instance Matching with COMA++. BTW Workshops, 2007, 28-37
2. Halevy, A. Why Your Data Won't Mix Queue, ACM, 2005, 3, 50-58
3. Shvaiko, P.; Euzenat, J. Ontology Matching: State of the Art and Future Challenges. IEEE Transactions on Knowledge and Data Engineering, 2011, 99
4. Zaiss, K.; Schlueter, T.; Conrad, S. Instance-Based Ontology Matching Using Different Kinds of Formalisms. Proceedings of the International Conference on Semantic Web Engineering, Oslo, Norway, July, 2009, 29-31

Ontology Alignment based on Instances using Hybrid Genetic Algorithm

Alex Alves, Kate Revoredo, Fernanda Baião

Research and Practice Group in Information Technology (NP2Tec)
Department of Applied Informatics – Federal University of the State of Rio de Janeiro (UNIRIO)
{alex.alves,katerevoredo,fernanda.baiao}@uniriotec.br

Abstract. The popularity of Ontology favored the appearance of several Ontologies to the same domain, thereby increasing the need of alignment techniques. In scenarios where ontologies comprising instances, the knowledge embedded in these instances can be useful to improve alignment. This paper extends a hybrid evolutionary approach, which applies a local optimization method, by taking instances into account in order to reduce premature convergence and, consequently, improve the quality of the resulting ontology alignment.

1 Introduction

Ontology is an explicit formal specification of the concepts in a domain and relations among them. In the existence of two or more ontologies for the same domain, there is the need of finding the correspondences between them. This task is known as *ontology alignment* [Shvaiko and Euzenat 2007] and is accomplished by evaluating the elements of the two ontologies, trying to find the best pair of corresponding elements. Finding this correspondence is not an easy task, especially in domains with many concepts and relations, where scalable approaches maybe necessary. In [Acampora et. al. 2012], ontology alignment has been formulated as an optimization problem applying a genetic algorithm with local search heuristics.

On the other hand, in scenarios where ontologies contain instances, the knowledge embedded in these instances can be useful to improve the alignments. Therefore, the ontology elements that may be considered for the alignment comprise its concepts, relations, or instances [Shvaiko and Euzenat 2007]. The alignment approach proposed in [Acampora et. al. 2012] considered only the first two elements. In this paper, we delineate ideas towards extending their approach by also considering instances.

2 Ontology Alignment based on Instances and through Genetic Algorithms

Genetic algorithms try to solve an optimization problem (or search) by manipulating a population of potential solutions that reproduces the process of natural evolution. Specifically, they operate on encoded representations of the solutions, called chromosomes, an equivalent representation of an individual feature in nature. The evolution algorithm starts from a population of individuals randomly generated and creates successive generations. At each generation, a natural selection process takes place, providing a mechanism for selecting the best solution to survive. Each solution is evaluated by means of a fitness function and compared to other solutions in the population. The higher a fitness value of an individual is the greater will be its chances of surviving. When creating a new generation, the recombination of genetic material among individuals of a generation applies two operators: crossover (which exchanges portions between two randomly selected chromosomes) and mutation (which causes random alteration of the genes of one chromosome). The evolution algorithm terminates when some specified conditions are reached [Acampora et. al. 2012].

In our approach, as in [Acampora et. al. 2012], a genetic algorithm is applied to solve the ontology-alignment problem. A chromosome corresponds to a potential alignment between two given ontologies (O^1 and O^2), and is represented by an integer vector $A = (e_0, \ldots, e_{n-1})$ such that, when the i-th position of A has a value of e_j, this means the alignment between the e_i element from O^1 and the e_j element from O^2, that is, the correspondence (e_i, e_j). The length of A is given by the number of the elements of O^1. Figure 1 illustrates an example of a fictitious alignment between the elements from two ontologies in a car domain (where each double-ended arrow connects a pair of corresponding elements) and the chromosome representing this alignment.

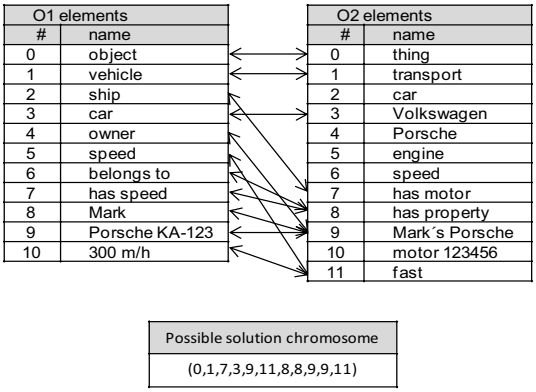


Fig. 1. A possible alignment between O_1 and O_2 ontologies
and its chromosome representation (adapted from [Acampora et al. 2012])

In order to take instances into account during the alignment problem, we propose an additional function. This function applies the concept of upPropagation [Massmann et al. 2011], in which the similarities between instances are propagated to their concepts when evaluating a possible solution. Moreover, our approach will initially adopt specific values for the genetic parameters, following the work of Souza [2012]: a selection rate of 50%, crossover probability of 80%, mutation probability of 10%, a 30% rate for reinsertion of best individuals, a 10% rate for reinsertion of the worst individuals, mortality of 5 generations, a local search frequency of every 100 generations and, finally, a 25% insertion neighborhood. By adopting this parameter value set, avoid solutions that persist for many generations, like super individuals or solutions very bad, is applied the concept of mortality in the population. Individuals that reach a certain age m are dropped from the new generation. Finally, we assume the existence of a reference alignment and predefined thresholds for precision, recall and F-measures as our stopping criteria.

3 Conclusion

Typically, ontologies are used by people, artificial agents and distributed applications that need to share domain information about a specific subject or area of knowledge. However, the creation of these ontologies is commonly performed in accordance with local needs and often without concern for reuse. In an ever-increasing frequent scenario where various ontologies for the same domain exist, alignment of them is a must, but still remains as a challenging problem. In many of these scenarios, instances may potentially bring extra information helping the alignment process, but are currently under-exploited in the literature, especially when combined with other approaches. In this paper, we propose to use instances to improve the alignment of ontologies through the use of a genetic algorithm combined with a local search heuristic to reduce premature convergence. Experiments are being performed to evaluate our proposal.

References

Acampora, G., Loia, V., Salerno, S. and Vitiello, A. (2012), “A hybrid evolutionary approach for solving the ontology alignment problem”, In: International Journal of Intelligent Systems, 27:189–216. doi:10.1002/int.20517.

Euzenat, J. and Shvaiko, P. (2007), “Ontology Matching”, Springer-Verlag, Berlin Heidelberg. 2007, X, 334 p. 67 illus. ISBN 978-3-540-49611-3.

Massmann, S.; Raunich, Salvatore; Aumueller, David; Arnold, Patrick; Rahm e Erhard. (2011)” Evolution of the COMA Match System”, OM-2011 (The Sixth International Workshop on Ontology Matching, October 24th, 2011, Bonn, Germany).

Souza, Jairo Francisco. (2012) “A heuristic approach single-objective for calibration in meta-ontology aligners”. Rio de Janeiro, 105p. PhD Thesis, Department of Computer Science, Pontifícia Universidade Católica do Rio de Janeiro.

Direct computation of diagnoses for ontology alignment*

Kostyantyn Shchekotykhin, Philipp Fleiss, Patrick Rodler, and Gerhard Friedrich

Alpen-Adria Universität, Klagenfurt, 9020 Austria
firstname.lastname@aau.at

Abstract. Modern ontology debugging methods allow efficient identification and localization of faulty axioms in an ontology. However, in many use cases such as ontology alignment the ontologies might include many conflict sets, i.e. sets of axioms preserving the faults, thus making ontology diagnosis infeasible. In this paper we present a debugging approach based on a direct computation of diagnoses that omits calculation of conflict sets. The evaluation results show that the approach is practicable and is able to identify a fault in adequate time.

1 Algorithm details and evaluation

Most of the modern debugging approaches apply the model-based diagnosis [3] and compute diagnoses using conflict sets CS, i.e. irreducible sets of axioms ax_i in an ontology $\mathcal{O}$ that preserve a fault. A user should modify at least all axioms of a diagnosis in order to be able to formulate the intended (target) ontology $\mathcal{O}_t$. The computation of the conflict sets can be done within a polynomial number of calls to the reasoner, e.g. by QUICKXPLAIN algorithm [2]. To identify a diagnosis of cardinality $|\mathcal{D}| = m$ the hitting set algorithm suggested in [3] requires computation of m conflict sets. In the use cases when an ontology is generated by an ontology matching system the number of conflict sets m can be large, thus making the ontology debugging practically infeasible.

In this paper we present two algorithms INV-HS-TREE and INV-QUICKXPLAIN, which inverse the standard model-based approach and compute diagnoses directly, rather than by means of conflict sets. INV-QUICKXPLAIN partitions the initial set of axioms a given faulty ontology into two equal subsets. The algorithm continues to partition the sets until it identifies that the set $\mathcal{D}'$ such that $\mathcal{O} \setminus \mathcal{D}'$ fulfills all requirements and its partitions are not. In further iterations the algorithm minimizes the $\mathcal{D}'$ by splitting it into sub-problems of the form $\mathcal{D} = \mathcal{D}' \cup \mathcal{O}_\Delta$, where $\mathcal{O}_\Delta$ contains only one axiom. In the case when $\mathcal{D}$ is a diagnosis and $\mathcal{D}'$ is not, the algorithm decides that $\mathcal{O}_\Delta$ is a subset of the sought diagnosis. Just as the original algorithm, INV-QUICKXPLAIN always terminates and returns either a diagnosis $\mathcal{D}$ or "no diagnosis". In order to enumerate all possible diagnoses we modified the HS-TREE algorithm [3] to accept diagnoses as node labels instead of conflict sets.

In the diagnosis discrimination settings [4] the ontology debugger acquires new knowledge by asking the user whether some axiom should be entailed by the target ontology $\mathcal{O}_t$ or not. Given the answer the algorithm can invalidate some of the diagnoses that are used as labels of tree nodes. Given such a node, INV-HS-TREE removes its label and places it to the list of open nodes. Moreover, the algorithm removes all the nodes of a subtree originating from this node. After all nodes with invalid labels are cleaned-up,

* This research is funded by Austrian Science Fund (Project V-Know, contract 19996).

the algorithm attempts to reconstruct the tree by reusing the remaining valid diagnoses. In the direct approach limiting the number of diagnoses used to compute a query to some reasonable number. e.g. $n = 10$ results in a small size of the search tree, thus, using less memory in comparison to the standard approach.

We evaluated the direct ontology debugging technique using aligned ontologies generated in the framework of OAEI 2011 [1]. These ontologies represent a real-world scenario in which a user generated ontology alignments by means of some (semi-)automatic tools. The Conference test suite we included 146 classifiable ontologies and computed 1, 9 and 30 diagnoses with both HS-TREE and INV-HS-TREE. For 133 ontologies both approaches were able to compute the required amount of diagnoses. In the experiment where only 1 diagnosis was requested, the direct approach outperforms the HS-TREE as it was expected. In the next two experiments the time difference between the approaches decreases. However, the direct approach was able to avoid a rapid increase of computation time for very hard cases. In the 13 cases HS-TREE was unable to find all requested diagnoses in each experiment. Within 2 hours the algorithm calculated only 1 diagnosis for `csa-conference-ekaw` and for `ldoa-conference-confof` it was able to find 1 and 9 diagnoses, whereas INV-HS-TREE required 9 sec. for 1, 40 sec. for 9 and 107 sec. for 30 diagnoses on average.

Moreover, in the first experiment we evaluated the efficiency of the interactive direct debugging approach applied to the 13 "hard" ontologies. We selected the target diagnosis randomly among all diagnoses that included only invalid alignments suggested by a system. The latter can be computed using the set of correct alignments provided by the organizers of OAEI 2011. In the experiment the used the Entropy scoring function [4] with prior fault probabilities of axioms corresponding to aliments set to $1 - v$, where v is the confidence value of the matcher. All axioms of the aligned ontologies were assumed to be correct and were assigned small probabilities. The debugging was then applied to the set of all alignments returned by a matcher. The experiment shows that the system was able to identify the target diagnosis efficiently requiring less than 4 sec. in 75% of all cases to compute a query. The system's performance decreased only in the cases when a reasoner required much time to verify the consistency of an ontology.

In the second scenario we applied the direct method to unsatisfiable and classifiable within 2 hours ontologies, generated for the Anatomy problem. The source ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ include 11545 and 4838 axioms correspondingly, whereas the size of the alignments varies between 1147 and 1461 axioms. The target diagnosis selection process was performed in the same way as in the first experiment. The results of the experiment show that the target diagnosis can be computed within 40 second in an average case. Moreover, INV-HS-TREE slightly outperformed HS-TREE.

References

1. Euzenat, J., Ferrara, A., van Hage, W.R., Hollink, L., Meilicke, C., Nikolov, A., Ritze, D., Scharffe, F., Shvaiko, P., Stuckenschmidt, H., Sváb-Zamazal, O., dos Santos, C.T.: Final results of the Ontology Alignment Evaluation Initiative 2011. In: Proceedings of the 6th International Workshop on Ontology Matching. pp. 1–29. CEUR-WS.org (2011)
2. Junker, U.: QUICKXPLAIN: Preferred Explanations and Relaxations for Over-Constrained Problems. In: Proc. 19th National Conference on Artificial Intelligence. pp. 167–172 (2004)
3. Reiter, R.: A Theory of Diagnosis from First Principles. Artificial Intelligence 32(1), 57–95 (1987)
4. Shchekotykhin, K., Friedrich, G., Fleiss, P., Rodler, P.: Interactive ontology debugging : two query strategies for efficient fault localization. Journal of Web Semantics 12, 88–103 (2012)

Measuring Semantic Similarity within Reference Ontologies to Improve Ontology Alignment

Valerie Cross and Pramit Silwal

Computer Science and Software Engineering Department,
Miami University, Oxford, OH 45056
crossv@muohio.edu

1 Mediating Matcher with Semantic Similarity

Some ontology alignment (OA) systems find an identical bridge concept in a reference ontology to which both the source and target concept can be mapped. Then a mapping between the two is produced. Using semantic similarity within a reference ontology can find more mappings than with only identical bridge concepts. A wide variety of semantic similarity measures were implemented within AgreementMaker [1] to use semantic similarity to evaluate OA mappings [2]. Initial results of enhancing AgreementMaker with a new matcher, the mediating matcher with semantic similarity (MMSS) in place of its mediating matcher (MM) are in [3]. Briefly, the MMSS uses the MM to first produce a set of mappings M_{ST} between source and target concepts with an exact match on the bridge concepts, i.e., $b_S = b_T$ as

$$M_{ST} = \{(s, t, mapSim_{SR} * mapSim_{TR}) \mid s \in O_S, b_S, b_T \in O_R, t \in O_T : \exists(s, b_S, mapSim_{SR},) \in M_{SR} \wedge \exists(t, b_T, mapSim_{TR},) \in M_{TR} \wedge b_S = b_T\}$$

M_{SR} contains mapping from the source O_S to the reference O_R using BSM^{lex} matcher and similarly for M_{TR} with O_T. U_S contains source concepts s from M_{SR}, which are not selected by the original MM and similarly U_T for the target concepts t.

$$U_S = \{s \mid s \in O_S : \exists(s, b_S, mapSim_{SI}) \in M_{SI} \wedge \nexists\, t \in O_T : (s, t, sim_{ST}) \in M_{ST}\}$$

For each (s, t) in U_S x U_T, semantic similarity between all their b_S and b_T are calculated, and the maximum is used in determining the enhanced mapping set as

$$E_{ST} = \{(s, t, agg(mapSim_{SR}, mapSim_{TR}, bridgeSim)) \mid s \in U_S, b_S, b_T \in O_R, t \in U_T : \exists(s, b_S, mapSim_{SR}) \in M_{SR} \wedge \exists\,(t, b_T, mapSim_{TR},) \in M_{TR} : bridgeSim = max_{b_S, b_T \in O_R}\, (semSim(b_S, b_T))\}.$$

Different *agg* operators are possible, but here minimum is used with the rationale that the final mapping between s and t is not any stronger than the weakest similarity between the pairs of concepts, (s,b_S), (t,b_T), and (b_S,b_T). Different semantic similarity measures can be used for *semSim*. The standard Lin semantic similarity measure is used with information content described as in [3]. An additional threshold value may be set to eliminate mappings in E_{ST} whose aggregated similarity falls below the threshold. $M_{ST} \cup E_{ST}$ is input to the linear weighted combination (LWC) operation

2 Experimental Results using OAEI 2011 Anatomy Track

To compare the MMSS to the MM in OAEI 2011 AgreementMaker configuration using its matchers and hierarchical LWCs, experiments are performed with the OAEI 2011 anatomy track and Uberon as the reference ontology. The results are shown in Table 1. At the 0.9 threshold level, the OAEI 2011 AgreementMaker configuration with MMSS (OAEI-MMSS) produced 2 more mappings than with MM (OAEI-MM), but no more correct mappings. Examining the mappings showed OAEI-MMSS found 3 new correct ones but lost 3 correct ones found by OAEI-MM. Further analysis suggests that the interaction among AgreementMaker's matchers, its local quality measures (LQM) used as weighting for its LWCs, and the hierarchical organization of its LWCs have subtle effects on the mappings eventually selected for the final result.

Table 1. OAEI 2011 AgreementMaker MM vs. its MMSS version

	Produced	Correct	Precision	Recall	F-measure
OAEI-MM	1439	1348	93.7	88.9	91.2
OAEI-MMSS -0.9	1441	1348	93.5	88.9	91.2
OAEI-MMSS-0.95	1441	1350	93.7	89.1	91.3
OAEI-MMSS-0.95, PSM kept	1443	1353	93.8	89.2	91.4

The OAEI 2011 AgreementMaker configuration hierarchically combines its Parametric String-based Matcher (PSM), Vector-based Multi-word Matcher (VMM) and Lexical Similarity Matcher (LSM) represented as LWC3(LWC1(LSM+MM) + LWC2(PSM + VMM)) using LQMs to weight each component. MMSS is substituted for MM. Other hierarchical combinations of matchers in experiments did not perform better than OAEI-MM. However, three different hierarchical combinations produced new mappings not found by either the OAEI-MM or OAEI-MMSS for a total of 9 new correct mappings. More work is needed to determine possible heuristics to be able to keep the lost 3 mappings and also retain the 9 new mappings. Examining LQM weighting showed LQMs for MMSS are usually higher than for the PSM or VMM; therefore, the MMSS dominates in the final results. The third table row shows going from 0.9 to 0.95 eliminates incorrect mappings to retain the lost mappings. The last row shows by keeping identical source-target mappings from the PSM, a higher F-measure is achieved, better than AgreementMaker's OAEI 2011 result.

References

1. Cruz, I. F., Stroe, C., Caimi, F., Fabiani, A., Pesquita, C., Couto, F. M., Palmonari, M.: Using AgreementMaker to Align Ontologies for OAEI 2011. Ontology Matching Workshop, International Semantic Web Conference (2011)
2. Cross, V. and Hu, X: Using Semantic Similarity in Ontology Alignment. OM Workshop, 10th Int. Semantic Web Conference (ISWC 2011), Bonn Germany (2011)
3. Cross, V., Silwal, P., and Morell, D: Using a Reference Ontology with Semantic Similarity in Ontology Alignment. International Conference on Biomedical Ontologies (ICBO), July 22 – 25, Graz, Austria (2012)

Thesaurus Mapping: A Challenge for Ontology Alignment?

Dominique Ritze and Kai Eckert

Mannheim University Library
dominique.ritze,eckert@bib.uni-mannheim.de

Thesauri are hierarchical knowledge organization systems commonly used in libraries to categorize and index publications. While sometimes referred to as so-called lightweight ontologies [4], they actually fundamentally differ from ontologies in several aspects. Nevertheless, as thesauri are actively used, constantly maintained and improved, they offer an interesting background knowledge for semantic applications. This year, we reinstantiated the OAEI library track[1], i.e., we provide the ontology matching community with the challenge to create alignments between thesauri. First, we aim for interesting insights into the differences between ontologies and thesauri. Second, we try to further integrate existing thesauri by means of new alignments which leads to better search experiences within library systems. From 2007 to 2009, there has already been a library track in the OAEI [1]. They focused on matching thesauri describing the same topic but at a different level of granularity. For our track, we selected two very comparable thesauri with topical overlaps. To make sure that the created alignments are indeed used, we work closely together with the maintaining institutions. We apply the following two thesauri:

STW: The Thesaurus for Economics (STW) provides vocabulary on any economic subject: more than 6,000 standardized subject headings (in English and German) and 19,000 additional keywords. The entries are richly interconnected by 16,000 broader/narrower and 10,000 related relations. The vocabulary is maintained on a regular basis by ZBW German National Library of Economics – Leibniz Centre for Economics[2]. The thesaurus is available in SKOS [3].

TheSoz: Similar to the STW, the Thesaurus for the Social Sciences (TheSoz) serves as a crucial instrument for indexing documents and research information in the social sciences. Overall, it contains about 12,000 keywords, from which 8,000 are standardized subject headings (in English, German and French) and 4,000 additional ones. The thesaurus is owned and maintained by GESIS - Leibniz Institute for the Social Sciences[3]. The thesaurus is available in SKOS [5].

The matching results are evaluated by means of a reference alignment which has been manually created by domain experts in 2006 [2]. It has not been adapted or further developed after its initial creation. Hence, it does not cover changes of the thesauri. Within the reference alignment, concepts are aligned to more than one concept

[1] http://web.informatik.uni-mannheim.de/oaei-library/2012/
[2] http://zbw.eu/index-e.html
[3] http://www.gesis.org/en/home/

(*n:m* mapping). All in all, the alignment contains 2,839 exact matches and 1,450 subsumptions. Other generated correspondences will be evaluated by domain experts as well. It is planned to extend the reference alignment on the basis of manually evaluated matching results, if the quality is sufficient to justify the effort.

The participating matchers in OAEI are currently developed for (OWL) ontology matching. As a starting point for them, we provide an OWL version of the thesauri. Therefore, the SKOS predicates are mapped to RDF/OWL as follows:

SKOS	**RDF/OWL**
`skos:concept`	`owl:class`
`skos:prefLabel`, `skos:altLabel`	`rdfs:label`
`skos:scopeNote`, `skos:notation`	`rdfs:comment`
`skos:related`	`rdfs:seeAlso`
`skos:narrower`	`rdfs:superClassOf`
`skos:broader`	`rdfs:subClassOf`

There are several issues with such a mapping: First and foremost, a `skos:concept` is not a class. Concepts sometimes represent classes, like COMMODITIES, but there are other concepts that clearly represent instances, like GERMANY. The mapping of the broader/narrower relationships is likewise problematic. In the STW, the narrower path COMMODITIES → METALS → METAL PRODUCTS → RAZOR is found. All metals are commodities too, but metal products like a razor only consist of metal, but are no metal. And last, the expressiveness of SKOS regarding different types of labels, additional descriptive notes and general concept relations are lost in RDF/OWL.

Thus, the question arises to which degree the current matching systems are hampered by these oversimplifications and semantic inconsistencies. We indeed hope that specialized SKOS matchers will join the challenge and that they outperform the generic ontology matchers. This way, the library track can contribute to the integration of thesauri in real world applications. As a side-effect, we would like to raise the discussion, how thesauri relate to ontologies and which role they might play in the Semantic Web.

References

1. Antoine Isaac, Lourens van der Meij, Shenghui Wang, and Henk Matthezing. Results of the OAEI 2007 Library Thesaurus Mapping Track. Technical report, VU Amsterdam, 2007.
2. Philipp Mayr and Vivien Petras. Building a Terminology Network for Search: The KoMoHe Project. In *Proc. of the Int. Conference on Dublin Core and Metadata Applications*, pages 177 – 182, 2008.
3. Joachim Neubert. Bringing the "Thesaurus for Economics" on to the Web of Linked Data. In *Proc. of the WWW Workshop on Linked Data on the Web (LDOW)*, 2009.
4. Michael Uschold and Michael Gruninger. Ontologies and semantics for seamless connectivity. *SIGMOD Rec.*, 33(4):58–64, 2004.
5. Benjamin Zapilko, Johann Schaible, Philipp Mayr, and Brigitte Mathiak. TheSoz: A SKOS Representation of the Thesaurus for the Social Sciences. *Semantic Web – Interoperability, Usability, Applicability*. accepted.

Matching Geospatial Ontologies

Heshan Du[1], Natasha Alechina[1], Mike Jackson[1], Glen Hart[2]

[1] University of Nottingham
[2] Ordnance Survey of Great Britain

In recent years, multiple geospatial ontologies have been developed for a wide range of different spatial databases. In addition, the development of volunteered geographic information both challenges and provides opportunities to the traditional authenticated geospatial information. Though volunteered geographic information is typically not as reliable and structured as the authenticated geospatial information, it often reflects changes in the real world more quickly and contains richer information related to human activity [1]. It is therefore desirable to link the corresponding information from disparate geospatial information sources, allowing users to use them synergistically. Aligning disparate geospatial ontologies is an essential element to realizing this.

We propose a new semi-automatic method to align geospatial ontologies, based on coherence and consistency checking in description logic, as well as domain experts' knowledge. We evaluate it on real world data and compare it to two state of the art ontology mapping systems, CODI [2] and LogMap [3]. By a geospatial ontology we mean an ontology which contains both definitions of geospatial concepts in its TBox and facts about geospatial individuals in its ABox. When designing our approach, we assume that the TBox is not very large, but contains concepts which are more ambiguous, compared to for example biomedical ontologies. We also assume that geospatial individuals have geometry and location information. In common with other approaches, we use additional disjointness axioms to improve the quality of mapping. Since they are not part of the original ontology and may be wrong, we treat generated disjointness axioms as assumptions retractable by users. We treat original ontology axioms as correct and not retractable. Given two geospatial ontologies, our method has two main steps: generating assumptions and calculating a consistent and coherent assumption set (CAS) which contains the mapping.

Step 1: Retractable assumptions include disjointness axioms and mapping axioms. For TBoxes, disjointness axioms are generated for sibling classes. Initial mapping axioms between TBoxes are generated by stating equivalence of atomic concepts with identical names. Initial mapping axioms between ABoxes are generated based on three criteria: location, lexical labelling, and cardinality of mapping (one-to-one or one-to-many). We ensure that the geospatial instances from different sources are first represented at the same scale and using the same coordinate reference system scaling and transforming the input data as necessary. Given two instances, if their geometries are not spatially disjoint, we first generate a candidate 'sameAs' axiom for them. (When dealing with polygon geometries, the geometry checking is based on spatial disjointness, rather than shapes or sizes of geometries or their percentages of overlapping, because two

corresponding geospatial individuals may be represented differently in different datasets, and the representations may be of different geometry accuracy levels.) Then, each correspondence will be checked lexically. If the labels of the instances cannot be matched, we remove the correspondence. After that, the mapping will go through cardinality checking. In the case that several instances are mapped to the same instance, we change 'sameAs' relation to 'partOf' relation in the corresponding axioms. The geometry, lexical and cardinality checking are all necessary, since different geospatial individuals may share the same label or the same location in an ontology, and a same geospatial individual may be represented as a whole in one ontology, whilst as several parts of it in the other.

Step 2: Two ontologies are aligned by calculating a CAS with respect to them. We use Pellet [4] to check consistency and coherence of overall information. While inconsistency or incoherence exists, minimal inconsistent or incoherent assumption sets (MIAs) will be calculated and visualized clearly, allowing domain experts to correct them, until a CAS is obtained. We decide against automatic fixing of MIAs since none of the methods give entirely reliable results.

The method is implemented as a system called GeoMap. We evaluate it using the Ordnance Survey of Great Britain (OSGB) Buildings and Places ontology [5] and the OpenStreetMap (OSM) controlled vocabularies [6], which are representatives of formal and informal geospatial ontologies respectively. The data used in evaluation is available at `http://www.cs.nott.ac.uk/~hxd/GeoMap.html`. GeoMap, CODI [2] and LogMap [3] are employed to align the OSGB Buildings and Places ontology and the OSM ontology, extended with additional disjointness of siblings axioms. Based on manual evaluation, the precision rates of GeoMap, CODI and LogMap terminology mappings are 89%, 76% and 70% respectively. CODI generates 5 more correct mapping axioms than GeoMap, whilst LogMap generates 11 less. In the GeoMap instance mapping, more than 95% correspondences are reasonable. The experimental result shows that, when aligning geospatial ontologies, using geometry or location information helps, and domain experts are indispensable.

References

1. Jackson, M.J., Rahemtulla, H., Morley, J.: The Synergistic Use of Authenticated and Crowd-Sourced Data for Emergency Response. In: 2nd International Workshop on Validation of Geo-Information Products for Crisis Management (VALgEO). (2010)
2. Niepert, M., Meilicke, C., Stuckenschmidt, H.: A Probabilistic-Logical Framework for Ontology Matching. In: American Association for Artificial Intelligence. (2010)
3. Jiménez-Ruiz, E., Grau, B.C.: LogMap: Logic-Based and Scalable Ontology Matching. In: International Semantic Web Conference (1). (2011) 273–288
4. Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y.: Pellet: a Practical OWL-DL Reasoner. Web Semantics: Science, Services and Agents on the World Wide Web **5** (June 2007) 51–53
5. Hart, G., Dolbear, C., Kovacs, K., Guy, A.: Ordnance Survey Ontologies. http://www.ordnancesurvey.co.uk/oswebsite/ontology (2008)
6. OpenStreetMap: The Free Wiki World Map. http://www.openstreetmap.org (2012)

Leveraging SNOMED and ICD-9 cross mapping for semantic interoperability at a RHIO

Hari Krishna Nandigam MBBS, MSHI[1]; Vishwanath Anantharaman MBBS, MS[2]; James Heiman, MS[3]; Meir Greenberg, MD[2]; Michael Oppenheim, MD[2]

[1] Allscripts- NSLIJ Information System, New York, USA
[2] North Shore LIJ Health System, New York, USA
[3] Long Island Patient Information Exchange, New York, USA

Abstract: The HEAL10-NY grant, funded by the New York State Government, is an outcome of the state's effort to integrate hospitals, clinics and laboratories in New York. This paper discusses the mapping strategies adopted for the semantic and process interoperability between two hospital systems that have been implemented at two different clinical settings using standard vocabularies. Allscripts EMR is being used at ante/prenatal clinics to document ante/prenatal care and General Electricity's Centricity Perinatal (CPN) is being used at labor and delivery units in the hospital. Both applications differ in their content usage, software/hardware architectures, and platforms. Clinical information from these prenatal clinics/hospitals is transmitted after every visit to the RHIO, which acts as a clinical data repository and an aggregator of clinical information. At the time of admission for labor and delivery, the aggregated data is pushed to the inpatient labor and delivery system. This paper describes the usage of the recursive functions to traverse the SNOMED hierarchies to match the granularity levels of the Allscripts EMR application as well as SNOMED→ICD-9 crosswalk tables.

Keywords: Health Efficiency and Affordability Law for New Yorkers Phase 10 (HEAL10-NY) grant, Regional Health Information Organization (RHIO), recursive queries

Background: Lack of communication between healthcare providers at the hospitals, labs, and clinics is a common problem that leads to poor care coordination, and may result in low quality, or inefficient, patient care. [1] While most healthcare communities are able to achieve semantic interoperability at varying degrees, full interoperability is not accomplished until process interoperability is achieved. Semantic interoperability ensue that the various systems exchange and understand this clinical and/or administrative data without ambiguity. [4] However, unless the exchanged information is interpreted, aggregated and is presented to humans to facilitate better understanding of the patient's clinical condition, all the five rights for patient safety (the right patient, the right drug, the right dose, at the right time, with the right method of administration) cannot be ensured. This process interoperability will undoubtedly minimize unnecessary, or irrelevant, interventions on the patients. It will also coordinate work flow and enable the business processes at the healthcare facility.

Story Board Presentation: This section describes a typical case of high risk pregnant woman:

Before HEAL-10 Project: *A pregnant woman visits a private physician group practice for an antenatal checkup during her first trimester. Her initial visit to group practice didn't reveal any abnormalities. During her 18th gestational week, her primary obstetrician gives her a paper requisition to make an appointment at North Shore Long Island Jewish (NSLIJ) hospital for ultrasound. The performing obstetrician from NSLIJ calls her primary obstetrician and reads out the interpretation of the ultrasound. At month 6, her obstetrician at group practice diagnoses her with hypertension, and refers her to a specialist at Maternal & Fetal Medicine Unit within NSLIJ to examine her prenatal history and put her on a treatment program. Later, the patient visits the group practice a few times for a follow up. She also gets her lab work done at a private lab in Long Island. In month 9, the patient is admitted to NSLIJ for labor pains. The on-call obstetrician has no clue of how she was treated for hypertension during her antenatal period unless he/she calls the patient's primary obstetrician. Added to this, it becomes a challenge to get full information of this high risk pregnant woman who attended multiple clinics and/or hospitals/group practices elsewhere.*

After HEAL-10 Project: *The high risk pregnant woman was seen at a group practice and NSLIJ during her antenatal period. She got her lab work done at a private lab on Long Island. Her antenatal history and treatment were all recorded and stored at RHIO. She goes to NSLIJ for labor pains. The registration system at NSLIJ triggers a query to pull her antenatal history and treatment from RHIO. The system at RHIO summarizes her antenatal history and sends it to the system hosted at the OBGYN unit. The on-call obstetrician at NSLIJ can read through her antepartum record and can now make better decision and provide better care to this patient.*

The HEAL-10 project is a grant funded by New York State Department of Health (DOH) and Dormitory Authority of the State of New York (DASNY) to develop health information technology (Health IT) and restructure health care delivery system[5]. As part of HEAL Phase 10, NSLIJ partnered with LIPIX (a RHIO on Long Island)[6], and a number of physician practices, to develop 'A Patient Centered Medical Home Model for High Risk Obstetrics Using Electronic Medical Records'. High-risk pregnancy cases are most often complex and require a high coordination of care between the different care providers.

Most of the applications that record antepartum history essentially capture the clinical information recommended by the American College of Obstetrics and Gynecology (ACOG)[7]. Allscripts EHR, hosted at NSLIJ's Antenatal clinics, captures the obstetric history, the patients' pre-existing conditions, and the patients' medications. Few group practices in Long Island use GE's Centricity to document the antenatal care provided by their obstetricians.[8] External clinics send antenatal information to the RHIO. A Health information exchange (HIE) technology with a backend Centralized Data Repository (CDR) is required to ensure aggregation, normalization, and de-duplication of information from multiple systems that recorded the patients' information during the prenatal visits. LIPIX deployed a HIE product, 'HealthShare', developed by Intersystems Inc. HealthShare is a service oriented based vendor product operating on web services and is mostly used for exchange of health information across organizations.[9] Outpatient group practices send antenatal information asynchronously to HealthShare either as a batch process operating during night or as real time when a visit was completed. Data is sent as a combination of HL7 messages for ADT information and Continuity of Care Document (CCD). It is highly important for sending applications to codify clinical diagnoses (i.e. the problem list) to ICD-9 before transmitting CCD to HealthShare.

The data stored in HealthShare at LIPIX is 'pushed' to the GE Centricity Perinatal system when the patient presents to the hospital for labor and delivery. The GE Centricity Perinatal system is a specialized EMR developed by the obstetricians at the labor unit. This application does not have traditional ACOG problem list and was designed to capture only the essence of antenatal and critical clinical information. Further the system was built on a set of documentation templates that can be filled in based on patient's response. The goal of the project was to automatically populate the relevant fields in the admission history and physical examination templates with data coming from LIPIX. Critical to the process of importing data elements from external systems is mapping of the inbound data elements to CPN's data elements. Several thousands of clinical data elements have to be mapped to CPN's data elements and the challenge was to map concepts of higher granular to low granular concepts in CPN.

Mapping process: The first step in the mapping process was to identify the different sets of diagnoses in CPN. Some of the categories included Liver diseases, Gastro-intestinal diseases, Neurological disorders, Thyroid disorders etc. The basic idea was to leverage the SNOMED ontology and to treat each of the diagnoses categories in CPN as 'parent' concept and determine its child concepts in SNOMED database. We leveraged the 'is-a' relationship and recursively queried the SNOMED tables for all child concepts for each of the categories. In some instances there would be multiple categories for a given disease. E.g. Diabetic neuropathy could be classified as both a Diabetes Mellitus and Neurological disorder. After the SNOMED codes for the child concepts for each of the parent categories were determined, we cross-walked each of SNOMED concepts to their best ICD9 concept. We leveraged the SNOMED-ICD crosswalk provided by a commercial vendor Intelligent Medical Objects. The translated codes were then loaded into the HealthShare application to handle the ICD encoded prenatal clinical information coming from outpatient Allscripts EMR and mapped to concepts in inpatient CPN system. The methodology that we used allowed us to rapidly create mapping tables in an automated fashion. Since we leveraged existing SNOMED ontology, we eliminated the need for extensive testing of the mappings in itself. The discrete nature of the data allows us to leverage the exchanged information for various purposes including analytics and clinical decision support.

Conclusion and Future Work: Recursive functions are very useful in a hierarchical ontology database to query the child concepts. Use of recursive queries and automation of cross matching ontology tables becomes very significant for a regional health information organization that serves several hospitals in a region. Because hospitals deploys several hundreds of applications that have a dynamic life cycle, RHIOs need to constantly update their mapping tables in order to receive clinical information from sending application and cross map to meet the requirements of the receiving application. Automation of this process is quicker as well as less proned to errors. Most of the times a clinician review of mapping tables is indicated since a single error in cross map may result in dangerous outcomes.

References:

[1] S. Kripalani, F. LeFevre, C. O. Phillips, M. V. Williams, P. Basaviah, and D. W. Baker, "Deficits in communication and information transfer between hospital-based and primary care physicians: implications for patient safety and continuity of care," *JAMA*, vol. 297, no. 8, pp. 831–841, Feb. 2007.

[4] T. Benson, *Principles of Health Interoperability HL7 and SNOMED (Health Informatics) [Hardcover]*. Springer, 2009.

[5] "Request for Grant Applications - HEAL NY Phase 10 - Improving Care Coordination and Management Through a Patient Centered Medical Home Model Supported by an Interoperable Health Information Infrastructure." [Online]. Available: http://www.health.ny.gov/funding/rfa/inactive/0903160302/. [Accessed: 30-Jul-2012].

[6] "LIPIX - Long Island Patient Information Exchange Home." Available: http://www.lipix.org/. [Accessed: 17-Jul-2012].

[7] "ACOG - American Congress of Obstetricians and Gynecologists." Available: http://www.acog.org/. [Accessed: 30-Jul-2012].

[8] "GE Healthcare HIMSS." [Online]. Available: http://www.gehealthcare.com/centricity/. [Accessed: 19-Jul-2012].

[9] "InterSystems HealthShare - Revolutionizing Healthcare Informatics." Available: http://www.intersystems.com/healthshare/index.html. [Accessed: 17-Jul-2012].