Evaluating Mapping Repair Systems with Large Biomedical Ontologies

Ernesto Jiménez-Ruiz[1], Christian Meilicke[2], Bernardo Cuenca Grau[1], Ian Horrocks[1]

[1] Department of Computer Science, University of Oxford, Oxford UK,
[2] Research Group Data and Web Science, University of Mannheim, Germany

Abstract. In this paper we provide empirical evidence of the necessity of integrating mapping repair techniques within the ontology matching process, an aspect that is neglected in many ontology matching systems. We also evaluate the feasibility of using state-of-the-art mapping repair techniques in practice, such as those implemented in Alcomo and LogMap. A preliminary evaluation was conducted in the context of the Ontology Alignment Evaluation Initiative (OAEI) 2012. We extend this evaluation and report about the results in detail.

1 Introduction

OWL ontologies are extensively used in biomedical information systems. Prominent examples of biomedical ontologies are the National Cancer Institute Thesaurus (NCI) [14], the Foundational Model of Anatomy (FMA) [36] and the Systematized Nomenclature of Medicine Clinical Terms (SNOMED CT) [39]. These reference bio-medical ontologies, however, are being developed independently by different groups of experts and, as a result, they use different entity naming schemes in their vocabularies. For example, NCI defines the entity "Myocardium", whereas FMA uses the entity "Cardiac Muscle Tissue" to describe the muscles that surround and power the human heart. Thus, to integrate data among applications, it is crucial to establish correspondences (called mappings or alignments) between the entities of their respective ontologies.

In the last ten years, the Semantic Web and bio-informatics research communities have extensively investigated the problem of automatically computing correspondences between independently developed ontologies, usually referred to as the *ontology matching problem*. For example, the Ontology Alignment Evaluation Initiative[3] (OAEI) is an annual international campaign for the systematic evaluation of ontology matching systems [10, 9, 40, 28, 1]. The matching problems in the OAEI are organized in several tracks, with each track involving different kinds of test ontologies. For example, the *Large Biomed track* involves the matching of FMA, NCI and SNOMED CT.

OWL ontologies have well-defined semantics [5], however, many systems participating in the OAEI campaigns disregard the semantics of the input ontologies, and are thus unable to detect and repair logical errors (e.g. unsatisfiabilities) that follow from the union of the input ontologies and the mappings. Only the ontology matching systems S-Match [13], ASMOV [18], CODI [33, 17], KOSIMap [35], YAM++ [15] and

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

LogMap [20, 24] have implemented reasoning and repair techniques in the context of the OAEI. Furthermore, LogMap was the only system successfully applying such techniques in all tracks of the OAEI 2012 campaign [1].

In this paper, we focus on the evaluation conducted in the OAEI 2012 Large Biomed track and we provide an extension of the results presented in [1]. Concretely, we evaluate the feasibility and impact of integrating state-of-the-art mapping repair techniques, such as those implemented in Alcomo [29] or in LogMap, within the matching process.

2 Preliminaries

In this section, we first introduce the formal representation of ontology mappings. Next, we present the notions of mapping coherence and (approximate) mapping repair. Finally, we discuss how ontology matching systems are evaluated within the OAEI.

2.1 Representation of ontology mappings

Mappings are conceptualised as tuples of the form $\langle id, e_1, e_2, n, \rho \rangle$, with id a unique identifier for the mapping, e_1, e_2 entities in the vocabulary of the relevant ontologies, n a numeric confidence measure between 0 and 1, and ρ a relation between e_1 and e_2, typically subsumption (i.e., e_1 is more specific than e_2), equivalence (i.e., e_1 and e_2 are synonyms) or disjointness (i.e., no individual can be an instance of both e_1 and e_2) [8].

RDF Alignment [6] is the main format used in the OAEI campaign to represent mappings containing the aforementioned elements. Additionally, mappings are also represented as OWL 2 subclass, equivalence, and disjointness axioms [5]; mapping identifiers (id) and confidence values (n) are then represented as axiom annotations. Such a representation enables the reuse of the extensive range of OWL 2 reasoning infrastructure that is currently available. Note that alternative formal semantics for ontology mappings have been proposed in the literature (e.g., [4, 8, 31]).

2.2 Incoherent mappings and (approximate) mapping repair

The ontology $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$ resulting from the integration of $\mathcal{O}_1$ and $\mathcal{O}_2$ via a set of mappings $\mathcal{M}$, may entail axioms that do not follow from $\mathcal{O}_1$, $\mathcal{O}_2$, or $\mathcal{M}$ alone. In particular, classes that were satisfiable in $\mathcal{O}_1$ or $\mathcal{O}_2$ may become unsatisfiable w.r.t. $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$. A set of mappings that leads to such logical errors is referred to as *incoherent* [30].

Definition 1 (Mapping Incoherence). *A set of mappings $\mathcal{M}$ is incoherent with respect to $\mathcal{O}_1$ and $\mathcal{O}_2$, if there exists a class A in the signature of $\mathcal{O}_1 \cup \mathcal{O}_2$ such that $\mathcal{O}_1 \cup \mathcal{O}_2 \not\models A \sqsubseteq \bot$ and $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M} \models A \sqsubseteq \bot$.*

An incoherent set of mappings $\mathcal{M}$ can be fixed by removing mappings from $\mathcal{M}$. This process is referred to as *mapping repair* (or repair for short).

Definition 2 (Mapping Repair). *Let $\mathcal{M}$ be an incoherent set of mappings $\mathcal{M}$ w.r.t. $\mathcal{O}_1$ and $\mathcal{O}_2$. A set of mappings $\mathcal{R} \subseteq \mathcal{M}$ is a mapping repair for $\mathcal{M}$ w.r.t. $\mathcal{O}_1$ and $\mathcal{O}_2$ if $\mathcal{M} \setminus \mathcal{R}$ is coherent w.r.t. $\mathcal{O}_1$ and $\mathcal{O}_2$.*

An incoherent set of mappings can be repaired in many different ways. A trivial repair is $\mathcal{R} = \mathcal{M}$, since an empty set of mappings is obviously coherent (according to Definition 1). Nevertheless, the objective is to remove as few mappings as possible, which is consistent with the principle of minimal change in belief revision [11]. Such minimal repairs are typically referred to in the literature as *diagnosis* — a term coined by Reiter [34] and introduced to the field of ontology debugging in [37].

Definition 3 (Diagnosis). *Let $\mathcal{R}$ be a repair for $\mathcal{M}$ with respect to $\mathcal{O}_1$ and $\mathcal{O}_2$. $\mathcal{R}$ is a diagnosis if each $\mathcal{R}' \subset \mathcal{R}$ is not a repair for $\mathcal{M}$ with respect to $\mathcal{O}_1$ and $\mathcal{O}_2$.*

Standard justification-based ontology debugging techniques (e.g. [37, 38, 25, 41, 16, 22]) can be exploited to compute a repair (or a diagnosis) for an incoherent set of mappings. However, justification-based technologies do not scale when the number of unsatisfiabilities is large (a typical scenario in mapping repair problems [19]). To address this scalability issue, mapping repair systems usually compute an *approximate repair* using incomplete reasoning techniques. An approximate repair $\mathcal{R}^{\approx}$ does not guarantee that $\mathcal{M} \setminus \mathcal{R}^{\approx}$ is coherent, but it will (in general) reduce significantly the number of unsatisfiabilities caused by the original set of mappings $\mathcal{M}$.

2.3 Evaluation of ontology matching systems in the OAEI

The evaluation in the OAEI campaign is carried out automatically using the infrastructure developed within the EU project SEALS [42].[4] SEALS provides a repository to store test data (e.g. OAEI datasets) and an interface to consume this data and generate an output (e.g. set of mappings) following the accepted formats. OAEI participants have wrapped their systems according to the SEALS interface. Hence, OAEI systems are executed using the same workflow, which facilitates reproducibility of the experiments.

The quality of the mappings $\mathcal{M}$ computed by a matching system is often measured in terms of precision and recall with respect to a reference set of mappings (also called gold standard) $\mathcal{M}_{\mathrm{GS}}$. Precision (Pre) is defined as $|\mathcal{M} \cap \mathcal{M}_{\mathrm{GS}}|/|\mathcal{M}|$, while recall (Rec) is defined as $|\mathcal{M} \cap \mathcal{M}_{\mathrm{GS}}|/|\mathcal{M}_{\mathrm{GS}}|$. The F-score (F) combines precision and recall and is usually defined as their harmonic mean $(2 \times \text{Pre} \times \text{Rec})/(\text{Pre} + \text{Rec})$. The OAEI also evaluates the coherence of the computed mappings $\mathcal{M}$ with respect to the number of unsatisfiable classes obtained when reasoning with the input ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ together with $\mathcal{M}$. Additionally, computation times are also recorded.

3 The OAEI Large BioMed track

In this section we give an overview of the test data, participating systems and coherence results of the OAEI 2012 Large Biomed track.[5] The track involves the matching of FMA version 2.0 (78, 989 classes), NCI version 08.05d (66, 724 classes) and SNOMED CT Jan. 2009 version (306, 591 classes) and exploits the UMLS Metathesaurus [3] as the basis for the track's reference mappings (see [23, 21] for details). UMLS is the most

[4] Semantic Evaluation At Large Scale: `http://www.seals-project.eu`

[5] Large BioMed track: `http://www.cs.ox.ac.uk/isg/projects/SEALS/oaei/`

Table 1: Mapping coherence of the top 7 systems in the OAEI 2012 Large BioMed track

System	FMA-NCI		FMA-SNOMED CT		SNOMED CT-NCI	
	Unsat.	Ratio	Unsat.	Ratio	Unsat.	Ratio
LogMap$_{noe}$	9	0.01%	10	0.003%	$\geq$0	$\geq$0%
LogMap	9	0.01%	10	0.003%	$\geq$17	$\geq$0.005%
ServOMap	48,743	27%	273,242	71%	$\geq$313,643	$\geq$84%
ServOMapL	50,334	28%	99,726	26%	$\geq$314,939	$\geq$84%
GOMMA	5,574	4%	10,752	3%	$\geq$266,051	$\geq$71%
GOMMA$_{bk}$	12,939	9%	119,657	31%	$\geq$313,015	$\geq$84%
YAM++	50,550	29%	106,107	28%	$\geq$269,107	$\geq$72%

comprehensive effort for integrating independently-developed medical thesauri and ontologies, including FMA, NCI and SNOMED CT. Currently, the UMLS-based reference mappings only include subsumption and equivalence correspondences between classes.

The track consists of three matching problems: FMA-NCI, FMA-SNOMED CT and SNOMED CT-NCI; the gold standard is provided by their corresponding UMLS-based reference mappings; there are three tasks associated to each matching problem, each of which involves different fragments of the input ontologies. In this paper we focus on the tasks involving the matching of the whole FMA, NCI and SNOMED CT ontologies.

We have evaluated the coherence of the mappings obtained by the top systems in the OAEI 2012 Large BioMed track: LogMap, YAM++ [15], ServOMap [2], GOMMA [27] and some of their variants (LogMap$_{noe}$, GOMMA$_{bk}$ and ServOMapL). LogMap's default algorithm uses ontology modules to reduce the search space, while the variant LogMap$_{noe}$ does not rely on module extraction. GOMMA$_{bk}$, unlike GOMMA, exploits specialised background knowledge. ServOMapL is a light version of ServOMap with some features deactivated. Table 1 summarizes the obtained incoherence results, which have been obtained using the OWL 2 reasoner HermiT [32].[6] The table reports *(i)* number of unsatisfiable classes in $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$, where $\mathcal{M}$ represents the mappings computed by a given system, and *(ii)* the ratio of unsatisfiable classes over the total number of classes. The best results were obtained by LogMap and its variant LogMap$_{noe}$; LogMap$_{noe}$ managed to additionally detect some unsatisfiable classes that were missed by LogMap due to the fact that they fell outside the computed modules. The mappings computed by the other matching systems led to a huge number of unsatisfiable classes. For example, 71% of the classes in the integration of FMA and SNOMED CT via ServOMap mappings are unsatisfiable.

4 Mapping repair using Alcomo and LogMap

Alcomo and LogMap implement different techniques to repair incoherent mappings. In the evaluation conducted in this paper, both Alcomo and LogMap are configured to use incomplete reasoning. Thus, given two ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ and a set of mappings

[6] In the case of SNOMED CT-NCI no OWL 2 reasoner could succeed in classifying the integrated ontology via mappings [19], so we used the OWL 2 EL reasoner ELK [26] instead to provide a lower bound on the number of unsatisfiable classes.

$\mathcal{M}$ between them, Alcomo and LogMap compute an approximate repair $\mathcal{R}^{\approx}$ such that $\mathcal{M} \setminus \mathcal{R}^{\approx}$ is almost coherent and only leads to a (relatively) small number of unsatisfiable classes. Next we present an overview of the Alcomo and LogMap mapping repair techniques, the interested readers please refer to [29, 20, 24] for a full description of these systems. Note that LogMap was originally implemented as an ontology matching system, however, it can also operate as a stand-alone mapping repair system. From now on we will refer to LogMap's repair module as LogMap-Repair. Alcomo, unlike LogMap, has specifically been designed to repair incoherent mappings.

4.1 The Alcomo mapping repair system

Alcomo implementes two reasoning components. One component is a pattern-based reasoning technique that is incomplete for detecting all minimal incoherent mapping subsets. However, this approach will detect a large amount of conflicting pairs of mappings. The basic idea is to first classify both $\mathcal{O}_1$ and $\mathcal{O}_2$ using an OWL 2 reasoner. Given two mapping axioms $A_1 \equiv C_2 \in \mathcal{M}$ and $B_1 \equiv D_2 \in \mathcal{M}$ with A_1 and B_1 defined in $\mathcal{O}_1$ and C_2 and D_2 defined in $\mathcal{O}_2$, Alcomo checks if $\mathcal{O}_1 \models A_1 \sqsubseteq B_1$ and $\mathcal{O}_2 \models C_2 \sqsubseteq \neg D_2$. If this is the case, it can be concluded $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M} \models C_2 \sqsubseteq \bot$, i.e., C_2 is unsatisfiable in the ontology integrated via $\mathcal{M}$. Thus, the mapping set $\{A_1 \equiv C_2, B_1 \equiv D_2\}$ is incoherent. This basic idea is extended and four patterns are defined that take subsumption and equivalence mappings between classes and properties into account.

Depending on the configuration of Alcomo, these techniques can be accompanied with complete reasoning techniques that are built on the classical black-box approaches for repairing ontologies (e.g. [38, 25, 41, 16]). The basic idea of such a combined approach is to compute a preliminary superset of a solution based on the incomplete reasoning techniques. This intermediate result is then checked with complete reasoning techniques and further reduced if required. If complete reasoning techniques are activated, it can be guaranteed that Alcomo generates a coherent mapping set by removing $\mathcal{R}$ from $\mathcal{M}$. Moreover, $\mathcal{R}$ will be a diagnosis. Without activating complete reasoning techniques, Alcomo computes an approximate repair $\mathcal{R}^{\approx}$ and it cannot guarantee the coherence of the output mapping set. The approximate repair $\mathcal{R}^{\approx}$, however, will always be a subset (never a superset) of the diagnosis.

Mappings are usually annotated with confidences. Thus, the quality of a diagnosis can be defined in terms of the aggregated confidence of $\mathcal{R}$. An intuitive idea is to remove mapping sets with less aggregated confidence. Alcomo aims to solve two interconnected problems at the same time: *(i)* the reasoning problem of detecting and repairing incoherent mappings, and *(ii)* the optimization problem of taking confidences into account in the appropriate way. With respect to the optimization problem, two different types of diagnosis have been defined. A global optimal diagnosis is introduced as the diagnosis that removes as less confidence as possible. If all correspondences are weighted equally with respect to their (positive) confidence values, a global optimal diagnosis will be a diagnosis that is a smallest diagnosis in numbers of correspondences. This type of diagnosis is, however, computed by an exhaustive search algorithm, and thus it is not feasible to compute the global optimal diagnosis for large repair problems.

The second type of a diagnosis is called a local optimal diagnosis. Such a diagnosis can be constructed by a simple greedy approach starting with an empty mapping set

Algorithm 1 Alcomo's algorithm with local optimal diagnosis & incomplete reasoning

Input: $\mathcal{O}_1, \mathcal{O}_2$: input ontologies; $\mathcal{M}$: input mappings
Output: $\mathcal{M}'$: output mappings; $\mathcal{R}^{\approx}$: approximate mapping repair.

1: $\mathcal{M}' := \emptyset$
2: $\mathcal{R}^{\approx} := \emptyset$
3: classify $\mathcal{O}_1$ and $\mathcal{O}_2$
4: $C := \mathsf{ConflictPairs}(\mathcal{O}_1, \mathcal{O}_2, \mathcal{M})$
5: **for each** $m \in \mathcal{M}$ **do** ▷ iterate over $\mathcal{M}$ in descending order with respect to confidences
6:     $coh := \mathsf{true}$
7:     **for each** $m' \in \mathcal{M}'$ **do**
8:         **if** $(m', m) \in C$ **then**
9:             $coh := \mathsf{false}$
10:             **break**
11:         **end if**
12:     **end for**
13:     **if** $coh = \mathsf{true}$ **then** $\mathcal{M}' := \mathcal{M}' \cup \{m\}$
14:     **else** $\mathcal{R}^{\approx} := \mathcal{R}^{\approx} \cup \{m\}$
15: **end for**
16: **return** $\langle \mathcal{M}', \mathcal{R}^{\approx} \rangle$

$\mathcal{M}'$ that is extended step by step by adding mappings from $\mathcal{M}$. These mappings are ordered with respect to the confidence values starting with the highest confidence. Each time a mapping is added to $\mathcal{M}'$, the coherence is checked via a combination of pattern-based and complete reasoning techniques. If $\mathcal{M}'$ becomes incoherent, the mapping is not added to $\mathcal{M}'$. The resulting diagnosis $\mathcal{R} = \mathcal{M} \setminus \mathcal{M}'$ is also a minimal hitting set [34] over all conflicts, however, in general it is not a smallest confidence weighted diagnosis. A proof for this claim and a detailed explanation of an improved variant of this algorithm can be found in Section 6.1 in [29].

Both algorithms can be executed with complete reasoning activated or deactivated. In the latter case, only those logical errors that can be detected by the pattern-based reasoning approach are taken into account. In our experiments we have applied Alcomo in the setting that aims to compute a local optimal diagnosis using incomplete pattern-based reasoning techniques only. The corresponding pseudocode is shown in Algorithm 1. In Step 4 the pattern-based reasoning techniques described above are used to compute a set of conflicting pairs of mappings. Each of these pairs is incoherent with respect to $\mathcal{O}_1$ and $\mathcal{O}_2$. Note that most of the computational effort is dedicated to this (preprocessing) step. The remaining part of the algorithm requires no further reasoning and it is only required to check whether a certain combination of two mappings appears as a previously computed conflict pair.

4.2 The LogMap-Repair system

Algorithm 2 shows the pseudocode of the algorithm implemented by LogMap-Repair. Steps 1 and 2 initialise the output mappings $\mathcal{M}'$ with the input mappings $\mathcal{M}$ and the repair set $\mathcal{R}^{\approx}$ with the empty set. Note that LogMap-Repair splits equivalence mappings into two equivalent subsumption mappings. LogMap-Repair encodes the input

Algorithm 2 LogMap-Repair algorithm based on Horn propositional reasoning

Input: $\mathcal{O}_1, \mathcal{O}_2$: input ontologies; $\mathcal{M}$: input mappings
Output: $\mathcal{M}'$: output mappings; $\mathcal{R}^{\approx}$: approximate mapping repair.

1: $\mathcal{M}' := \mathcal{M}$
2: $\mathcal{R}^{\approx} := \emptyset$
3: $\langle \mathcal{P}_1, \mathcal{P}_2 \rangle := \mathsf{PropEncoding}(\mathcal{O}_1, \mathcal{O}_2)$
4: **for each** $C \in \mathsf{OrderedVariables}(\mathcal{P}_1 \cup \mathcal{P}_2)$ **do**
5:     $\mathcal{P}_C := \mathcal{P}_1 \cup \mathcal{P}_2 \cup \mathcal{M}' \cup \{\mathsf{true} \rightarrow C\}$
6:     $\langle sat, \mathcal{M}_\perp \rangle := \mathsf{DowlingGallier}(\mathcal{P}_C)$
7:     **if** $sat = \mathsf{false}$ **then**
8:         $Rep := \emptyset$
9:         $rep\_size := 1$
10:         **repeat**
11:             **for each** subset $\mathcal{R}_C$ of $\mathcal{M}_\perp$ of size rep_size **do**
12:                 $sat := \mathsf{DowlingGallier}(\mathcal{P}_C \setminus \mathcal{R}_C)$
13:                 **if** $sat = \mathsf{true}$ **then** $Rep := Rep \cup \{\mathcal{R}_C\}$
14:             **end for**
15:             $rep\_size := rep\_size + 1$
16:         **until** $Rep \neq \emptyset$
17:         $\mathcal{R}_C :=$ element of Rep with minimum aggregated confidence.
18:         $\mathcal{M}' := \mathcal{M}' \setminus \mathcal{R}_C$
19:         $\mathcal{R}^{\approx} := \mathcal{R}^{\approx} \cup \mathcal{R}_C$
20:     **end if**
21: **end for**
22: **return** $\langle \mathcal{M}', \mathcal{R}^{\approx} \rangle$

ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ as Horn propositional theories $\mathcal{P}_1$ and $\mathcal{P}_2$ (Step 3) and exploits this encoding to subsequently detect unsatisfiable classes in an efficient and sound way during the repair process. The theory $\mathcal{P}_1$ (resp. $\mathcal{P}_2$) consists of the following Horn rules:

– A rule $A \rightarrow B$ for all distinct classes A, B such that A is subsumed by B in $\mathcal{O}_1$ (resp. in O_2); subsumption relations can be determined using either an OWL 2 reasoner, or syntactically (in an incomplete way).
– Rules $A_i \wedge A_j \rightarrow \mathsf{false}$ ($1 \leq i < j \leq n$) for each disjointness axiom of the form $DisjointClasses(A_1, \ldots, A_n)$.
– A rule $A_1 \wedge \ldots \wedge A_n \rightarrow B$ for each subclass or equivalence axiom having the intersection of $A_1, \ldots A_n$ as subclass expression and B as superclass.

In Step 4, propositional variables in $\mathcal{P}_1$ (resp. in $\mathcal{P}_2$) are ordered such that a variable C in $\mathcal{P}_1$ (resp. in $\mathcal{P}_2$) comes before D whenever D is subsumed by C in $\mathcal{O}_1$ (resp. in $\mathcal{O}_2$). This is a well-known repair strategy: subclasses of an unsatisfiable class are also unsatisfiable and hence before repairing an unsatisfiable class one first needs to repair its superclasses. Satisfiability of a propositional variable C is determined by checking satisfiability of the propositional theory $\mathcal{P}_C$ consisting of *(i)* the rule $(\mathsf{true} \rightarrow C)$; *(ii)* the propositional representations $\mathcal{P}_1$ and $\mathcal{P}_2$; and *(iii)* the current set of output mappings $\mathcal{M}'$ (seen as propositional implications).

Algorithm 3 Evaluation of Alcomo and LogMap-Repair

Input: $\mathcal{O}_1$, $\mathcal{O}_2$: input ontologies; $\mathcal{M}_{\mathrm{GS}}$: reference mappings; MS: an ontology matching system

Compute mappings $\mathcal{M}$ (**I**) between $\mathcal{O}_1$ and $\mathcal{O}_2$ using system MS
Store matching time (**II**)
Compute F-score (**III**) of $\mathcal{M}$ with respect to $\mathcal{M}_{\mathrm{GS}}$
Get unsatisfiable classes of $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$ (**IV**) using a reasoner
Compute approximate repair $\mathcal{R}^{\approx}$ (**V**) using Alcomo system ▷ See Algorithm 1
Store repair time (**VI**)
Compute F-score (**VII**) of $\mathcal{M} \setminus \mathcal{R}^{\approx}$ with respect to $\mathcal{M}_{\mathrm{GS}}$
Get unsatisfiable classes of $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M} \setminus \mathcal{R}^{\approx}$ (**VIII**) using a reasoner
Compute approximate repair $\mathcal{R}^{\approx}$ (**IX**) using LogMap-Repair system ▷ See Algorithm 2
Store repair time (**X**)
Compute F-score (**XI**) of $\mathcal{M} \setminus \mathcal{R}^{\approx}$ with respect to $\mathcal{M}_{\mathrm{GS}}$
Get unsatisfiable classes of $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M} \setminus \mathcal{R}^{\approx}$ (**XII**) using a reasoner

LogMap-Repair implements the classical Dowling-Gallier algorithm for propositional Horn satisfiability [7, 12]. LogMap-Repair's implementation of Dowling-Gallier's algorithm also records all mappings potentially involved in an unsatisfiability. Thus, a call to Dowling-Gallier returns a satisfiability value sat and, optionally, the (overestimated) set of conflicting mappings $\mathcal{M}_\perp$ (see Steps 6 and 12). An unsatisfiable class C is repaired by discarding conflicting mappings for C (Lines 8 to 19). Thus, subsets $\mathcal{R}_C$ of $\mathcal{M}_\perp$ of increasing size are then identified until a repair is found (Steps 10-16).[7] Thus, LogMap-Repair does not compute a diagnosis for the unsatisfiable class C but rather the repairs of smallest size. If several repairs of a given size exist, the one with the lowest aggregated confidence is selected according to the confidence values assigned to mappings (Step 17). Finally, Steps 18 and 19 update the output mappings $\mathcal{M}'$ and the approximate mapping repair $\mathcal{R}^{\approx}$ by extracting and adding $\mathcal{R}_C$, respectively.

Algorithm 2 ensures that $\mathcal{P}_1 \cup \mathcal{P}_2 \cup \mathcal{M}' \cup \{\mathtt{true} \rightarrow C\}$ is satisfiable for each C occurring in $\mathcal{P}_1 \cup \mathcal{P}_2$. The propositional encoding of $\mathcal{O}_1$ and $\mathcal{O}_2$ is, however, incomplete and hence the algorithm does not ensure satisfiability of each class in $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}'$. Nevertheless, as shown in Section 5, the number of unsatisfiable classes remaining after computing an approximate repair $\mathcal{R}^{\approx}$ is typically small.

5 Evaluation

In this section we evaluate the feasibility of integrating the Alcomo and LogMap-Repair systems within the ontology matching process. For each of the matching problems of the OAEI 2012 Large BioMed track and for each of the top matching systems in this track (see Section 3) we have conducted the evaluation in Algorithm 3. The Roman numbers refer to measurements that are stored during the evaluation. We have run the evaluation using the SEALS interface in a high performance server with 16 CPUs and allocating 15 Gb RAM.

[7] The size of $\mathcal{M}_\perp$ and $\mathcal{R}_C$ are in practice manageable, and thus the complexity of performing Step 11 in Algorithm 2 is not critical.

Table 2: Mapping repair in the FMA-NCI problem.

System	Matching Results OAEI 2012				Repair with Alcomo				Repair with LogMap			
	I	II	III	IV	V	VI	VII	VIII	IX	X	XI	XII
	$\lvert\mathcal{M}\rvert$	t (s)	F	Unsat.	$\lvert\mathcal{R}^{\approx}\rvert$	t (s)	F	Unsat.	$\lvert\mathcal{R}^{\approx}\rvert$	t (s)	F	Unsat.
ServOMap	4,932	204	0.819	48,743	97	321	0.820	9	115	21	0.819	9
ServOMapL	5,400	251	0.841	50,334	126	342	0.841	9	166	24	0.839	9
GOMMA	5,686	217	0.839	5,574	123	321	0.839	15	142	20	0.838	9
GOMMA$_{\text{bk}}$	6,330	231	0.837	12,939	184	341	0.836	29	259	33	0.833	9
YAM++	5,476	1,304	0.862	50,550	116	324	0.862	10	141	21	0.861	9
Average	5,565	441	0.840	33,628	129	330	0.840	14	165	24	0.838	9

Table 3: Mapping repair in the FMA-SNOMED CT problem.

System	Matching Results OAEI 2012				Repair with Alcomo				Repair with LogMap			
	I	II	III	IV	V	VI	VII	VIII	IX	X	XI	XII
	$\lvert\mathcal{M}\rvert$	t (s)	F	Unsat.	$\lvert\mathcal{R}^{\approx}\rvert$	t (s)	F	Unsat.	$\lvert\mathcal{R}^{\approx}\rvert$	t (s)	F	Unsat.
ServOMap	12,642	532	0.770	273,242	829	2,672	0.749	0	2,640	284	0.721	0
ServOMapL	13,210	517	0.794	99,729	975	2,779	0.767	0	2,953	274	0.752	0
GOMMA	11,648	1,994	0.291	10,752	463	2,840	0.293	0	618	245	0.291	0
GOMMA$_{\text{bk}}$	25,660	1,893	0.708	119,657	1,726	2,809	0.698	1,363	5,295	314	0.678	0
YAM++	14,088	23,900	0.765	106,107	783	2,800	0.760	0	3,461	262	0.720	0
Average	15,450	5,767	0.666	121,897	955	2,780	0.653	273	2,993	276	0.632	0

Elements in $\mathcal{M}$ and $\mathcal{R}^{\approx}$ (**I**, **V** and **IX**) represent subsumption mappings. As in Table 1, the unsatisfiable classes (**IV**, **VIII** and **XII**) in the FMA-NCI and FMA-SNOMED CT matching problems have been computed using the HermiT reasoner, while in the SNOMED CT-NCI problem we have provided a lower bound using the ELK reasoner.

Tables 2-4 shows the result of the conducted evaluation using Algorithm 3. The results, which suggest that Alcomo and LogMap-Repair scale and produce very good results in practice, can be summarized as follows:

i the computed (approximate) repairs are not aggressive and the average size of the repairs ranges from 5% (Alcomo) to 11% (LogMap-Repair) of the input mappings,
ii the repair process, although it requires an (appreciable) additional computation time, does not represent a bottleneck in the matching process,
iii the impact on the F-score is (on average) negligible,[8] and
iv the incoherence of the repaired mapping sets has been significantly reduced in all test cases and completely removed in some of them.

Regarding the comparison between Alcomo and LogMap-Repair, Tables 2-4 also show that LogMap-Repair is 10 to 15 times faster compared to Alcomo, although Alcomo runtimes are slightly less affected by different mapping inputs; Alcomo is less aggressive and its repairs involve (in general) a smaller number of mappings, nevertheless LogMap-Repair results are better in terms of mapping coherence; finally the impact on the F-score is (on average) better in Alcomo.

[8] The computed (approximate) repairs have, in general, a negative impact on the recall which is compensated with an increase of the precision.

Table 4: Mapping repair in the SNOMED CT-NCI problem.

System	Matching Results OAEI 2012				Repair with Alcomo				Repair with LogMap			
	I	II	III	IV	V	VI	VII	VIII	IX	X	XI	XII
	$\vert\mathcal{M}\vert$	t (s)	F	Unsat.	$\vert\mathcal{R}^{\approx}\vert$	t (s)	F	Unsat.	$\vert\mathcal{R}^{\approx}\vert$	t (s)	F	Unsat.
ServOMap	24,924	654	0.664	$\geq$313,643	937	3,223	0.663	$\geq$35	1,671	276	0.666	$\geq$296
ServOMapL	27,928	738	0.678	$\geq$314,939	1,076	2,917	0.677	$\geq$2,055	1,656	314	0.679	$\geq$1,241
GOMMA	27,386	1,820	0.606	$\geq$266,051	1,903	2,947	0.603	$\geq$2,085	2,949	303	0.607	$\geq$37
GOMMA$_{bk}$	34,090	1,940	0.635	$\geq$313,015	2,720	3,098	0.638	$\geq$30,583	5,003	435	0.641	$\geq$1
YAM++	28,206	30,155	0.680	$\geq$269,107	757	2,964	0.679	$\geq$0	1,049	305	0.680	$\geq$0
Average	28,507	7,061	0.653	$\geq$295,351	1,479	3,030	0.652	$\geq$6,952	2,465	326	0.655	$\geq$315

6 Conclusions

In the paper we have pointed out that many ontology matching systems participating in the OAEI campaign do not implement or reuse methods to assess the coherence of the generated mappings. As a consequence, a large number of classes become unsatisfiable when reasoning with the matched ontologies together with the mappings. We have applied Alcomo and LogMap-Repair systems on the data sets and mapping results of the OAEI 2012 Large Biomed track to support two claims regarding the application of (approximate) mapping repair techniques: *(i)* it is feasible with respect to robustness and runtimes, and *(ii)* it has a significant impact on the quality of the mappings with respect to their logical coherence.

Our results clearly support both claims and should encourage ontology matching system developers to use Alcomo and LogMap-Repair, or to develop their own repair techniques. On the one hand, Alcomo and LogMap-Repair have been successfully applied to all data sets and matching systems. LogMap-Repair requires in all cases less time to compute a repair than the necessary time to compute the mappings; while Alcomo's times, although slower than LogMap-Repair's, are in many cases in line with the required matching time. On the other hand, Alcomo and LogMap-Repair reduced significantly the incoherence of the input mappings, and hence increasing their quality. Furthermore, the F-score stays relatively stable when applying the repairs.

The results also suggest that Alcomo and LogMap-Repair could complement each other. LogMap-Repair is more efficient in terms of runtimes and mapping coherence while Alcomo is less aggressive (i.e. removes less mappings even in those cases where the same mapping coherence results are achieved) and its impact in the F-score is smaller. Future work will involve the design and development of a repair algorithm combining the techniques implemented in Alcomo and LogMap-Repair.

Finally, we believe that our evaluation is not only beneficial for ontology matching system developers. It also serves as a good basis to compare ontology and mapping repair systems, in terms of efficiency and completeness, in a challenging scenario as the one exposed in the OAEI Large Biomed track. We highly encourage developers of such systems to compare their results against the results presented in this paper. The relevant data sets and mappings are available online at `http://www.cs.ox.ac.uk/isg/projects/SEALS/oaei/`.

Acknowledgements

The research presented in this paper was financed by the Seventh Framework Program (FP7) of the European Commission under Grant Agreement 318338, the Optique project. Horrocks, Jiménez-Ruiz, and Cuenca Grau were also partially supported by the EPSRC projects ExODA, LogMap and Score!

References

1. Aguirre, J., Eckert, K., Euzenat, J., Ferrara, A., van Hage, W.R., Hollink, L., Meilicke, C., Nikolov, A., Ritze, D., Scharffe, F., Shvaiko, P., Sváb-Zamazal, O., Trojahn, C., Jiménez-Ruiz, E., Cuenca Grau, B., Zapilko, B.: Results of the Ontology Alignment Evaluation Initiative 2012. In: Ontology Matching Workshop (2012)
2. Ba, M., Diallo, G.: Large-scale biomedical ontology matching with ServOMap. IRBM 34(1), 56 – 59 (2013), Special issue on digital technologies for healthcare
3. Bodenreider, O.: The unified medical language system (UMLS): integrating biomedical terminology. Nucleic Acids Research 32, 267–270 (2004)
4. Borgida, A., Serafini, L.: Distributed Description Logics: Assimilating information from peer sources. J. Data Sem. 1, 153–184 (2003)
5. Cuenca Grau, B., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P.F., Sattler, U.: OWL 2: The next step for OWL. J. Web Sem. 6(4), 309–322 (2008)
6. David, J., Euzenat, J., Scharffe, F., Trojahn, C.: The Alignment API 4.0. Semantic Web 2(1), 3–10 (2011), `http://alignapi.gforge.inria.fr/format.html`
7. 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)
8. Euzenat, J.: Semantic precision and recall for ontology alignment evaluation. In: Int'l Joint Conf. on Artif. Intell. (IJCAI). pp. 348–353 (2007)
9. Euzenat, J., Meilicke, C., Stuckenschmidt, H., Shvaiko, P., Trojahn, C.: Ontology alignment evaluation initiative: Six years of experience. J. Data Sem. 15, 158–192 (2011)
10. Euzenat, J., Shvaiko, P.: Ontology matching. Springer (2007)
11. Gaerdenfors, P.: Belief revision: An introduction. In: Belief Revision. Cambridge University Press (1992)
12. Gallo, G., Urbani, G.: Algorithms for testing the satisfiability of propositional formulae. J. Log. Prog. 7(1), 45–61 (1989)
13. Giunchiglia, F., Yatskevich, M., Shvaiko, P.: Semantic Matching: Algorithms and implementation. J. Data Sem. 9, 1–38 (2007)
14. Golbeck, J., Fragoso, G., Hartel, F.W., Hendler, J.A., Oberthaler, J., Parsia, B.: The National Cancer Institute's Thésaurus and Ontology. J. Web Sem. 1(1), 75–80 (2003)
15. Hoa Ngo, D., Bellahsene, Z.: YAM++ A combination of graph matching and machine learning approach to ontology alignment task. J. Web Sem. 16 (2012)
16. Horridge, M., Parsia, B., Sattler, U.: Laconic and precise justifications in OWL. In: Int'l Sem. Web Conf. (ISWC). pp. 323–338 (2008)
17. Huber, J., Sztyler, T., Noessner, J., Meilicke, C.: CODI: Combinatorial optimization for data integration: results for OAEI 2011. In: Ontology Matching Workshop (2011)
18. Jean-Mary, Y.R., Shironoshita, E.P., Kabuka, M.R.: Ontology matching with semantic verification. J. Web Sem. 7(3), 235–251 (2009)
19. 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)

20. Jiménez-Ruiz, E., Cuenca Grau, B.: LogMap: Logic-based and Scalable Ontology Matching. In: Int'l Sem. Web Conf. (ISWC). pp. 273–288 (2011)
21. Jiménez-Ruiz, E., Cuenca Grau, B., Horrocks, I.: Exploiting the UMLS Metathesaurus in the Ontology Alignment Evaluation Initiative. In: Exploiting Large Knowledge Repositories (E-LKR) Workshop (2012)
22. 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)
23. 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)
24. Jiménez-Ruiz, E., Cuenca Grau, B., Zhou, Y., Horrocks, I.: Large-scale interactive ontology matching: Algorithms and implementation. In: European Conf. on Artif. Intell. (ECAI). pp. 444–449 (2012), `http://code.google.com/p/logmap-matcher/`
25. 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)
26. Kazakov, Y., Krötzsch, M., Simancik, F.: Concurrent classification of EL ontologies. In: Int'l Sem. Web Conf. (ISWC). pp. 305–320 (2011)
27. Kirsten, T., Gross, A., Hartung, M., Rahm, E.: GOMMA: a component-based infrastructure for managing and analyzing life science ontologies and their evolution. J. Biomed. Sem. 2 (2011)
28. Meilicke, C., Svab-Zamazal, O., Trojahn, C., Jiménez-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, 1208.3148 (2012), `http://arxiv.org/abs/1208.3148v1`
29. Meilicke, C.: Alignment Incoherence in Ontology Matching. Ph.D. thesis, University of Mannheim (2011)
30. Meilicke, C., Stuckenschmidt, H.: Incoherence as a basis for measuring the quality of ontology mappings. In: Ontology Matching Workshop (2008)
31. Meilicke, C., Stuckenschmidt, H., Tamilin, A.: Reasoning support for mapping revision. J. Log. Comput. 19(5) (2009)
32. Motik, B., Shearer, R., Horrocks, I.: Hypertableau reasoning for description logics. J. Artif. Intell. Res. (JAIR) 36, 165–228 (2009)
33. Niepert, M., Meilicke, C., Stuckenschmidt, H.: A probabilistic-logical framework for ontology matching. In: AAAI Conf. on Artif. Intell. pp. 1413–1418 (2010)
34. Reiter, R.: A theory of diagnosis from first principles. Artificial Intelligence 32, 57–95 (1987)
35. Reul, Q., Pan, J.Z.: KOSIMap: Use of description logic reasoning to align heterogeneous ontologies. In: Description Logics Workshop (2010)
36. Rosse, C., Mejino Jr., J.: A reference ontology for biomedical informatics: the Foundational Model of Anatomy. J. Biomed. Informatics 36(6), 478–500 (2003)
37. Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In: Int'l Joint Conf. on Artif. Intell. (IJCAI). pp. 355–362 (2003)
38. Schlobach, S., Huang, Z., Cornet, R., van Harmelen, F.: Debugging incoherent terminologies. J. Autom. Reasoning 39(3) (2007)
39. Schulz, S., Cornet, R., Spackman, K.A.: Consolidating SNOMED CT's ontological commitment. Applied Ontology 6(1), 1–11 (2011)
40. Shvaiko, P., Euzenat, J.: Ontology Matching: State of the art and future challenges. IEEE Trans. Knowledge and Data Eng. (2012)
41. 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)
42. Wrigley, S.N., Garcia-Castro, R., Nixon, L.J.B.: Semantic evaluation at large scale (SEALS). In: The 21st World Wide Web Conf., WWW (Companion Volume). pp. 299–302 (2012)