Improving Automatically Created Mappings using Logical Reasoning

Christian Meilicke[1], Heiner Stuckenschmidt[1], Andrei Tamilin[2]

[1] University of Mannheim, Germany
[2]ITC-irst and University of Trento, Italy

Abstract. A lot of attention has been devoted to heuristic methods for discovering semantic mappings between ontologies. Despite impressive improvements, the mappings created by these automatic matching tools are still far from being perfect. In particular, they often contain wrong and redundant mapping rules. In this paper we present an approach for improving such mappings using logical reasoning in the context of Distributed Description Logics (DDL). Our method is orthogonal to the matching algorithm used and can therefore be used in combination with any matching tool. We explain the general idea of our approach informally using a small example and present the results of experiments conducted on the OntoFarm Benchmark which is part of the Ontology Alignment Evaluation challenge.

1 Motivation

The problem of semantic heterogeneity is becoming more and more pressing in many areas of information technologies. The Semantic Web is only one area where the problem of semantic heterogeneity has lead to intensive research on methods for semantic integration. The specific problem of semantic integration on the Semantic Web is the need to not only integrate data and schema information, but to also provide means to integrate ontologies, rich semantic models of a particular domain. There are two lines of work connected to the problem of a semantic integration of ontologies:

- The (semi-) automatic detection of semantic relations between ontologies (e.g., [9, 6, 11, 12, 7]).
- The representation and use of semantic relations for reasoning and query answering (e.g., [14, 10, 5, 3, 2]).

So far, work on representation of and reasoning with mappings has focussed on mechanisms for answering queries and using mappings to compute subsumption relationships between concepts in the mapped ontologies. These methods always assumed that the mappings used are manually created and of high quality (in particular consistent). In this paper we investigate logical reasoning about mappings that are not assumed to be perfect. In particular, our methods can be used to check (automatically created) mappings for formal and conceptual consistency and determine implied mappings that have not explicitly been represented. We investigate such mappings in the context of Distributed Description Logics [1, 13], an extension of traditional description logics

with mappings between concepts in different T-boxes. The functionality described in this paper will become more important in the future because more and more ontologies are created and need to be linked. For larger ontologies the process of mapping will not be done completely by hand, but will rely on or will at least be supported by automatic mapping approaches. We see our work as a contribution to semi-automatic approaches for creating mappings between ontologies where possible mappings are computed automatically and then corrected manually making use of methods for checking the formal and conceptual properties of the mappings.

In previous work we have proposed a number of formal properties of mappings in Distributed Description Logics that we consider useful for judging the quality of a set of mappings [16]. In this paper, we refine and extend this work in several directions.

Debugging of mappings We propose a process for (semi-)automatically debugging automatically created mappings making use of some of the properties mentioned above. In particular we use the notion of mapping consistency to detect problems caused by the mappings. For each potential problem, we determine the minimal set of mapping rules responsible for the problem (minimal conflict set). For each conflict set, we try to identify which mapping rule is incorrect and remove it form the mapping.

Implementation On top of the DRAGO reasoning system [15] we built a prototype of mapping debugger for computing minimal conflict sets with respect to an inconsistency caused by a mapping as well as some heuristics for automatic repairing of an inconsistent mapping. We further added a minimization functionality for computing minimal mapping sets from redundant ones.

Experiments We tested the approach using the OntoFarm data set, a set of several rich OWL ontologies describing the domain of conference management systems [17]. We used the CtxMatch matching tool to automatically create mappings between each of the ontologies. We further automatically determined problems (in particular unsatisfiable concepts) created by the mapping and tried to fix them automatically using the debugging process proposed in this paper. In the concluding step of the experimental study, we tried to compute for each mapping its logically-equivalent minimal version.

The structure of the paper is as follows. We start with a brief recall of basic definitions of Distributed Description Logics and explanations of the reasoning mechanisms. Then we describe the intuitions of our debugging/minimization approaches using a small example. Finally, we report on some preliminary experimental evaluation of the techniques proposed in this paper and summarize the results.

2 Distributed Description Logic

Distributed Description Logic framework (DDL) is a formal tool for representing and reasoning with multiple ontologies pairwise linked by semantic mappings. In this section, we briefly recall some key definitions and properties of DDL relying on the original studies in [1, 13].

2.1 Syntax and Semantics

Given a set I of indexes, used to enumerate a set of ontologies, a *Distributed Description Logics* is then a collection $\{\mathcal{D}L_i\}_{i\in I}$ of Description Logics. Each ontology i is formalized by a T-box $\mathcal{T}_i$ of $\mathcal{D}L_i$, so that the initial set of ontologies in DDL corresponds to a family of T-boxes $\mathcal{T} = \{\mathcal{T}_i\}_{i\in I}$. To distinguish the descriptions from various $\mathcal{T}_i$ in the family, DDL utilizes a prefix notation to pin descriptions to ontologies where they are considered in, e.g., $i : X$, $i : X \sqsubseteq Y$. Semantic relations between pairs of ontologies a represented in DDL by bridge rules. A *bridge rule* from i to j is an expression of the following two forms:

$i : X \xrightarrow{\sqsubseteq} j : Y$ – an *into-bridge rule*

$i : X \xrightarrow{\sqsupseteq} j : Y$ – an *onto-bridge rule*

where X and Y are concepts of ontologies $\mathcal{T}_i$ and $\mathcal{T}_j$ respectively. The derived bridge rule $i : X \xrightarrow{\equiv} j : Y$ can be defined as the conjunction of corresponding into- and onto-bridge rule.

Intuitively, the into-bridge rule $i : Bachelor \xrightarrow{\sqsubseteq} j : Student$ states that, from the j-th point of view the concept $Bachelor$ in i is more specific than its local concept $Student$. Similarly, the onto-bridge rule $i : ScientificEvent \xrightarrow{\sqsupseteq} j : Conference$ expresses the more generality relation.

A *distributed T-box* $\mathfrak{T} = \langle \mathcal{T}, \mathfrak{B}\rangle$ consists of a collection of T-boxes $\mathcal{T} = \{\mathcal{T}_i\}_{i\in I}$ and a collection of bridge rules $\mathfrak{B} = \{\mathfrak{B}_{ij}\}_{i\neq j\in I}$ between them.

The semantics of DDL is based on the key assumption that each ontology $\mathcal{T}_i$ in the family is *locally* interpreted by interpretation $\mathcal{I}_i$ on its *local* interpretation domain $\Delta^{\mathcal{I}_i}$. The semantic correspondences between heterogeneous local domains, e.g., the representations of a registration fee in US Dollars and in Euro, are modeled in DDL by a domain relation.

A *domain relation* r_{ij} represents a possible way of mapping the elements of $\Delta^{\mathcal{I}_i}$ into the domain $\Delta^{\mathcal{I}_j}$: $r_{ij} \subseteq \Delta^{\mathcal{I}_i} \times \Delta^{\mathcal{I}_j}$ such that r_{ij} denotes $\{d' \in \Delta^{\mathcal{I}_j} \mid \langle d, d'\rangle \in r_{ij}\}$; for any subset D of $\Delta^{\mathcal{I}_i}$, $r_{ij}(D)$ denotes $\bigcup_{d\in D} r_{ij}(d)$; and for any $R \subseteq \Delta^{\mathcal{I}_i} \times \Delta^{\mathcal{I}_j}$ $r_{ij}(R)$ denotes $\bigcup_{\langle d,d'\rangle\in R} r_{ij}(d) \times r_{ij}(d')$. For instance, if $\Delta^{\mathcal{I}_1}$ and $\Delta^{\mathcal{I}_2}$ are the representations of a registration fee in US Dollars and in Euro, then r_{12} could be a rate of exchange function, or some other approximation relation.

A *distributed interpretation* $\mathfrak{I} = \langle \{\mathcal{I}_i\}_{i\in I}, \{r_{ij}\}_{i\neq j\in I}\rangle$ of a distributed T-box $\mathfrak{T} = \langle \mathcal{T}, \mathfrak{B}\rangle$ consists of a family of local interpretations $\mathcal{I}_i$ on local interpretation domains $\Delta^{\mathcal{I}_i}$, one for each $\mathcal{T}_i$, and a family of domain relations r_{ij} between these local domains. A distributed interpretation $\mathfrak{I}$ is said to satisfy a distributed T-box $\mathfrak{T} = \langle \mathcal{T}, \mathfrak{B}\rangle$, written $\mathfrak{I} \models \mathfrak{T}$, if all T-boxes in $\mathcal{T}$ are satisfied

$$\mathfrak{I} \models \mathcal{T}_i, \text{ if } \mathcal{I}_i \models A \sqsubseteq B \text{ for all } A \sqsubseteq B \in \mathcal{T}_i$$

and all bridge rules in $\mathfrak{B}$ are satisfied:

$$\mathfrak{I} \models i : X \xrightarrow{\sqsubseteq} j : Y, \text{ if } r_{ij}(X^{\mathcal{I}_i}) \subseteq Y^{\mathcal{I}_j}$$
$$\mathfrak{I} \models i : X \xrightarrow{\sqsupseteq} j : Y, \text{ if } r_{ij}(X^{\mathcal{I}_i}) \supseteq Y^{\mathcal{I}_j}$$

Given a distributed T-box $\mathfrak{T} = \langle \mathcal{T}, \mathfrak{B} \rangle$, one can perform some basic Distributed DL inferences. A concept $i : C$ is *satisfiable* with respect to $\mathfrak{T}$ if there exist a distributed interpretation $\mathfrak{I}$ of $\mathfrak{T}$ such that $C^{\mathcal{I}_i} \neq \emptyset$. A concept $i : C$ is *subsumed* by a concept $i : D$ with respect to $\mathfrak{T}$ ($\mathfrak{T} \models i : C \sqsubseteq D$) if for every distributed interpretation $\mathfrak{I}$ of $\mathfrak{T}$ we have that $C^{\mathcal{I}_i} \subseteq D^{\mathcal{I}_i}$.

2.2 DDL Inference Mechanisms

Although both in DL and Distributed DL the fundamental reasoning services lay in verification of concepts satisfiability/subsumption within a certain ontology, in DDL, besides the ontology itself, the reasoning also depends on other ontologies that affect it through semantic mappings. This affection consist in the ability of bridge rules to *propagate the knowledge* across ontologies in form of subsumption axioms.

The simplest case illustrating the knowledge propagation in DDL is the following:

$$\frac{i : A \sqsubseteq B, \quad i : A \xrightarrow{\sqsupseteq} j : G,\ i : B \xrightarrow{\sqsubseteq} j : H}{j : G \sqsubseteq H} \tag{1}$$

In languages that support disjunction, the simplest propagation rule can be generalized to the propagation of subsumption between a concept and a disjunction of other concepts in the following way:

$$\frac{i : A \sqsubseteq B_1 \sqcup \ldots \sqcup B_n, \quad i : A \xrightarrow{\sqsupseteq} j : G,\ i : B_k \xrightarrow{\sqsubseteq} j : H_k\ (1 \leq k \leq n)}{j : G \sqsubseteq H_1 \sqcup \ldots \sqcup H_n} \tag{2}$$

The important property of the described knowledge propagation is that it is directional, i.e., bridge rules from i to j support knowledge propagation only from i towards j. It has been shown in [13] that adding the inference pattern (2) to existing DL tableaux reasoning methods lead to a correct and complete method for reasoning in DDL. This method has been implemented in the DRAGO DDL reasoner.

3 The Debugging Process

In this section we will explain the general idea of our approach for improving automatically created mappings based on reasoning about mappings in Distributed Description Logics using a simple example. In particular, we consider two ontologies in the domain of conference management systems, the same domain we did our experiments in. For each ontology, i and j, we only consider a single axiom, namely:

$$i : Author \sqsubseteq Person \qquad \text{and} \qquad j : Person \sqsubseteq \neg Authorization$$

These simple axioms that describe the concept of a person in two different ontologies – one stating that an author is a special kind of person and the other one stating that the concepts Person and Authorization (to access submitted papers) are disjoint concept – are enough to explain the important features of our approach. The approach consists of the following steps.

3.1 Mapping Creation

In the first step, we use any existing system for matching ontologies to create an initial set of mapping hypotheses. In particular, we are interested in mappings between class names, because these are the kinds of mappings that we can reason about using DDL framework. In order to support automatical repair of inconsistent mappings later on, the matching algorithm chosen should ideally not only return a set of mappings, but also a level of confidence in the correctness of a mapping. For the sake of simplicity, we assume that we use a simple string matching method that compares the overlap in concept names and computes a similarity value that denotes the relative size of the common substring[1]. Mappings are created based on a threshold for this value that we assume to be 1/3. Applying this method to the example will result in the following two mappings with corresponding levels of confidence:

$$i : Person \xrightarrow{\equiv} j : Person, 1.00$$
$$i : Author \xrightarrow{\equiv} j : Authorization, 0.46$$

We further assume that the mapping method also applies some structural heuristics to derive additional mappings and propagates the levels of confidence accordingly. For instance, the fact that $i : Person$ is a superconcept of $i : Author$ which is assumed to be equivalent to $j : Authorization$ may be used to derive the following mapping:

$$i : Person \xrightarrow{\sqsupseteq} j : Authorization, 0.46$$

In the same way, the fact that $i : Author$ is a subconcept of $i : Person$ and the fact that $i : Person$ is assumed to be equivalent to $j : Person$ may be used to the following addition mapping:

$$i : Author \xrightarrow{\sqsubseteq} j : Person, 1.00$$

We can easily see that the process has produced two incorrect mappings, namely the ones with a confidence of 0.46. It could be argued that it is easy to get rid of these incorrect mappings by raising the threshold to 0.5 for instance. This however is no sustainable solution to the problem, because there might be mappings with a level of confidence below 0.5 that are correct, on the other hand, there might still be incorrect mappings with a confidence of more than 0.5. Instead of relying on artificially set thresholds, we propose to analyze the impact of created mappings on the connected ontologies and to eliminate mappings that have a malicious influence.

3.2 Diagnosis

The mapping set described in the last step now serves as a basis for analyzing the effect of mappings and detecting malicious mappings. This process is similar to the well known concept of model-based diagnosis which has already successfully been applied to the task of detecting wrong axioms in single ontologies. Similar to existing approaches for diagnosing ontologies, our starting point are unsatisfiable concepts which

[1] of course we use more sophisticated methods in the real experiments

are interpreted as symptoms for which a diagnosis has to be computed. Compared to the general task of diagnosing ontologies, we are in a lucky position, because we have to deal with a much smaller set of potential diagnosis. In particular, we claim that the ontologies connected in the first step do not contain unsatisfiable concepts. If we now observe unsatisfiable concepts in the target ontology[2] and assuming that the ontologies themselves are correct, we know that they have to be caused by some mappings in the mapping set.

To illustrate this situation, we can have a look at our example again. Using existing techniques for reasoning in DDL, we can derive that the concept Authorization is globally unsatisfiable, i.e., $j : Authorization^{\mathcal{I}} = \emptyset$, because we have $Authorzation \sqsubseteq \neg Person$ and at the same time, we can infer $Authorization \sqsubseteq Person$. There are two reasons for this, namely:

$$j : Authorization^{\mathcal{I}_j} = r_{ij}(i : Author^{\mathcal{I}_i}) \subseteq r_{ij}(i : Person^{\mathcal{I}_i}) = j : Person^{\mathcal{I}_j}$$

and

$$j : Authorization^{\mathcal{I}_j} \subseteq r_{ij}(i : Person^{\mathcal{I}_i}) = j : Person^{\mathcal{I}_j}$$

Interpreting the inconsistency of the concept $j : Authorization$ as a symptom, we can now try to identify and repair the cause of this inconsistency. For this purpose, we compute irreducible conflict set for this symptom. Here an irreducible conflict set is a set of mappings that makes the concept unsatisfiable and has the additional property that removing a mapping from the set makes the concept satisfiable again. the arguments above it is easy to see that he have the following irreducible conflict sets:

$$\{i : Person \xrightarrow{\equiv} j : Person, i : Author \xrightarrow{\equiv} j : Authorization\}$$

and

$$\{i : Person \xrightarrow{\equiv} j : Person, i : Person \xrightarrow{\sqsupseteq} j : Authorization\}$$

In classical diagnosis, all conflict sets[3] are computed and the diagnosis is computed from these conflict sets using the hitting set algorithm. For the case of diagnosing mappings this is neither computationally feasible nor does it provide the expected result. In our example, the hitting set would consist of the mapping $i : Person \xrightarrow{\equiv} j : Person$ which, as we sill see later, is the only mapping that actually carries some correct information.

Our solution to the problem is to use an iterative approach that computes an often not minimal hitting set by determining one conflict set at a time and immediately fixing it in the way described in the next section. In our example, the algorithm will first detect the second conflict and fix it, afterwards, the method checks whether the concept $j : Authorization$ is still inconsistent. As this is the case, the second conflict set will be detected and fixed as well removing the problem.

[2] the formal semantics of DDL guarantees that the addition of mappings cannot lead to unsatisfiable concepts in the source ontology

[3] in classical diagnosis often only minimal conflict sets are considered

3.3 Heuristic Debugging

As mentioned above, the result of the diagnosis step is an irreducible conflict sets, in particular a set of mappings that make a concept unsatisfiable and with the additional property that removing one mapping from this set solves the problem in the sense that the concept becomes satisfiable. The underlying idea of our approach is now that unsatisfiable concepts are the result of wrong mappings. This means that each irreducible conflict set contains at least one mapping rule that does state a correct semantic relation between concepts and therefore should not be in the set of mappings. The goal of the debugging step is now to identify this malicious mapping and remove it from the overall mapping set. If we chose the right mapping for removal the quality of the overall mapping set should be improved, because a wrong mapping has been removed. In the case of our example, the first irreducible conflict set that will be considered consists of the following two mappings one of which we have to remove:

$$i : Person \xrightarrow{\equiv} j : Person, 1.00$$
$$i : Author \xrightarrow{\equiv} j : Authorization, 0.46$$

There are different ways now, in which a decision about the mapping to remove could be made. The easiest way is to use an interactive approach where the conflict sets are presented to a human user who decides which mapping should be removed. In our case, the user will easily be able to decide that the mapping $i : Author \xrightarrow{\equiv} j : Authorization$ is not correct and should be removed. In the second iteration, the following two mappings will be in the irreducible conflict set:

$$i : Person \xrightarrow{\equiv} j : Person, 1.00$$
$$i : Person \xrightarrow{\sqsupseteq} j : Authorization, 0.46$$

For this set the user will be able to see immediately that the second mapping should be removed, because it is not correct. This approach sound trivial, but in the presence of large mapping sets, providing the user with feedback about potential problems in terms of small conflict sets is of great help and often reveals problems that are hard to see when looking at the complete mapping set.

We can also try to further automate the debugging process by letting the system decide, which mapping rule to eliminate. In cases where the matching system already provides a measure of confidence, this is again quite simple, as we can simply remove the mapping rule with the lowest degree of confidence. In our case this is again the rule $i : Author \xrightarrow{\equiv} j : Authorization$ and removing it will lead to a better mapping set. It is not always possible, however, to rely on the confidence provided by the matching system, either because the system simply does not provide any or because the levels of confidence provided are not informative. In our experiments, we often had the situation where all mapping even though they were conflicting had a confidence of 100% attached. In this case, we have to think of a new way of ranking mappings. An approach that we used in our experiments that turned out to work quite well is to compute the semantic distance of the concept names involved using WordNet synsets. For the example above it is clear that this heuristic will also lead to an exclusion of the second

rule, because the class names in the first rule are equivalent and therefore have the least semantic distance possible. In cases where no distinction can be made using this heuristic, we have to switch back to the interactive mode and ask the user which mapping to remove. In any cases, the debugging step leaves us with a single mapping that does not create any inconsistencies. In order to get a complete set of correct mappings, we can now infer all additional mappings that follow from this one which leads us to the corrected final set of mappings in our case this final set if the following.

$$i : Person \xrightarrow{\equiv} j : Person, 1.00$$
$$i : Author \xrightarrow{\sqsupseteq} j : Person, 1.00$$

In summary, the process above is a way to improve the quality of automatically generated mapping sets by means of intelligent post-processing. Using formal properties of mappings and logical reasoning we are able to detect wrong mappings by analyzing their impact and tracking unwanted effects back to the mapping rules that caused them. In this our method is not yet another ontology matching method, but it is actually orthogonal to existing developments in the area of ontology matching as it can be applied to any set of mappings. The approach can be extended in several directions. First of all we can use symptoms other than concept satisfiability as a starting point for diagnosis. Further, we can use the method on joint sets of competing mappings created by different matching algorithms. This will help us to get a better coverage of the actual semantic relations and the trust in the quality of the different matching algorithms provides us with an additional criterion for selecting mappings to be discarded.

3.4 Minimization

A further improvement of the debugged mapping can be achieved by removing redundant mappings - mappings that logically follow from other mappings. In [16] we defined the notion of minimality of a mapping that we use in this context to remove redundant mappings. In the example for instance, the two mappings derived using structural heuristics do not really add new information to the system, because they can be derived from the two equivalence mappings that have been created first. In particular $i : Person \xrightarrow{\sqsupseteq} j : Authorization$, is redundant information, because:

$$i : Author^{\mathcal{I}_i} \subseteq i : Person^{\mathcal{I}_i} \tag{3}$$
$$\implies r_{ij}(Author^{\mathcal{I}_i}) \subseteq r_{ij}(Person^{\mathcal{I}_i}) \tag{4}$$
$$r_{ij}(Person^{\mathcal{I}_i}) = j : Person^{\mathcal{I}^j} \tag{5}$$
$$\implies r_{ij}(Author) \subseteq j : Person^{\mathcal{I}_j} \tag{6}$$

This means that for reasoning with automatically created mappings, we only have to take into account the equivalence mapping between the person concept in the two ontologies, because it is the basis for inferring the other one. For this reason, we remove all mappings that can be shown to be redundant in the sense that they can be derived from using other mappings from the set of mappings and only continue with the resulting minimal mapping set that still carries all the semantics of the complete set.

4 Experiments

In this section we report on some preliminary experimental evaluation of the mapping debugging/minimization techniques presented in the preceding sections. All the experiments have been conducted on the prototype of the debugger/minimizer implemented on top of the DRAGO DDL reasoner [15].

4.1 Experimental Setting

To perform experiments, we used a set of ontologies developed in the OntoFarm project [17] which are used as a part of Benchmark in Ontology Alignment Evaluation challenge.[4] In particular, we selected several ontologies modeling the domain of *conference organization*:

Ontology	Description Logics Expressivity	Number of classes	Number of properties
CMT	$\mathcal{ALCIF}(\mathcal{D})$	30	59
CONFTOOL	$\mathcal{SIF}(\mathcal{D})$	39	36
CRS	$\mathcal{ALCIF}(\mathcal{D})$	14	17
EKAW	$\mathcal{SHIN}$	73	33
PCS	$\mathcal{ALCIF}(\mathcal{D})$	24	38
SIGKDD	$\mathcal{ALCI}(\mathcal{D})$	51	28

Given this ontology test set, we apply the following experimental scenario. Using the CtxMatch matching tool [4], we automatically compute mappings between pairs of ontologies in the test set. Among the created mappings, we further identify those ones which are capable of producing unsatisfiable classes and therefore need to be debugged first. In the process of debugging, malicious bridge rules in mappings are automatically diagnosed and removed in accordance with the heuristic debugging discussed in Section 3. In the concluding step of the experimental study, we apply the minimization algorithm to compute for each mapping a logically-equivalent minimal set of bridge rules. Note that for those mappings which demand the debugging first the minimization is applied to their repaired descendants.

4.2 Results

The results of applying the heuristic debugging and minimization techniques to the automatically generated mappings are summarized in Table 1 and Table 2. More information about the test data and results can be obtained visiting the applications section of the DRAGO reasoner web page.[5]

During the debugging process we performed the following measurements: the initial amount of bridge rules in the mapping to be debugged, number of classes which become unsatisfiable due to the mapping, and finally the sets of bridge rules which are diagnosed as malicious and are automatically removed by the debugging algorithm. After the removal of malicious bridge rules, a mapping becomes repaired in a sense that it is not capable of producing unsatisfiability anymore. As shown in Table 1, the results of

[4] `http://nb.vse.cz/~svabo/oaei2006/`
[5] `http://sra.itc.it/projects/drago/applications.html`

Mapping	Bridge rules count	Unsatisfiable Classes count	Removed bridge rules count	Set of removed bridge rules
CMT-CONFTOOL	48	3	3	$CMT : Conference \xrightarrow{\sqsubseteq} CONFTOOL : Organization$ $CMT : Person \xrightarrow{\sqsupseteq} CONFTOOL : Poster$ $CMT : ProgramCommitteeChair \xrightarrow{\sqsubseteq} CONFTOOL : Event$
CMT-CRS	53	1	1	$CMT : Document \xrightarrow{\sqsupseteq} CRS : program$
CMT-EKAW	116	4	5	$CMT : Person \xrightarrow{\sqsupseteq} Flyer$ $CMT : Person \xrightarrow{\sqsupseteq} Multi-author\_Volume$ $CMT : Person \xrightarrow{\sqsupseteq} Proceedings$ $CMT : ProgramCommitteeChair \xrightarrow{\sqsubseteq} Social\_Event$ $CMT : Person \xrightarrow{\sqsupseteq} Conference\_Proceedings$
CONFTOOL-CRS	80	10	15	$CONFTOOL : University \xrightarrow{\sqsubseteq} CRS : event$ $CONFTOOL : Social\_event \xrightarrow{\sqsupseteq} CRS : program$ $CONFTOOL : Author \xrightarrow{\sqsubseteq} CRS : event$ $CONFTOOL : Person \xrightarrow{\sqsupseteq} CRS : event$ $CONFTOOL : Participant \xrightarrow{\sqsubseteq} CRS : event$ $CONFTOOL : Event \xrightarrow{\sqsupseteq} CRS : participant$ ...
CRS-CMT	53	2	3	$CRS : document \xrightarrow{\sqsupseteq} CMT : Acceptance$ $CRS : document \xrightarrow{\sqsupseteq} CMT : ProgramCommitteeChair$ $CRS : program \xrightarrow{\sqsupseteq} CMT : ProgramCommitteeChair$
CRS-CONFTOOL	80	27	30	$CRS : conference \xrightarrow{\sqsubseteq} CONFTOOL : Organization$ $CRS : person \xrightarrow{\sqsubseteq} CONFTOOL : Event$ $CRS : person \xrightarrow{\sqsupseteq} CONFTOOL : Poster$ $CRS : document \xrightarrow{\sqsubseteq} CONFTOOL : Event$ $CRS : author \xrightarrow{\sqsubseteq} CONFTOOL : Event$ $CRS : participant \xrightarrow{\sqsubseteq} CONFTOOL : Event$ $CRS : event \xrightarrow{\sqsupseteq} CONFTOOL : Person$ ...
PCS-CONFTOOL	45	5	5	$PCS : Conference \xrightarrow{\sqsubseteq} CONFTOOL : Organization$ $PCS : Report \xrightarrow{\sqsubseteq} CONFTOOL : Event$ $PCS : Report \xrightarrow{\sqsubseteq} CONFTOOL : Organization$ $PCS : PERSON \xrightarrow{\sqsupseteq} CONFTOOL : Poster$ $PCS : Accepted\_paper \xrightarrow{\sqsubseteq} CONFTOOL : Event$
PCS-EKAW	120	4	5	$PCS : PERSON \xrightarrow{\sqsupseteq} EKAW : Flyer$ $PCS : PERSON \xrightarrow{\sqsupseteq} EKAW : Multi-author\_Volume$ $PCS : PERSON \xrightarrow{\sqsupseteq} EKAW : Proceedings$ $PCS : Web\_site \xrightarrow{\sqsubseteq} EKAW : Event$ $PCS : PERSON \xrightarrow{\sqsupseteq} EKAW : Conference\_Proceedings$
SIGKDD-CMT	60	1	1	$SIGKDD : Program\_Committee \xrightarrow{\sqsupseteq} CMT : ProgramCommitteeChair$
SIGKDD-CONFTOOL	72	3	3	$SIGKDD : Conference \xrightarrow{\sqsubseteq} CONFTOOL : Organization$ $SIGKDD : Person \xrightarrow{\sqsupseteq} CONFTOOL : Poster$ $SIGKDD : Deadline\_Author\_notification \xrightarrow{\sqsubseteq} CONFTOOL : Person$
SIGKDD-CRS	57	1	1	$SIGKDD : Document \xrightarrow{\sqsupseteq} CRS : program$
SIGKDD-EKAW	127	4	5	$SIGKDD : Person \xrightarrow{\sqsupseteq} EKAW : Flyer$ $SIGKDD : Person \xrightarrow{\sqsupseteq} EKAW : Multi-author\_Volume$ $SIGKDD : Person \xrightarrow{\sqsupseteq} EKAW : Proceedings$ $SIGKDD : Deadline\_Author\_notification \xrightarrow{\sqsubseteq} EKAW : Person$ $SIGKDD : Person \xrightarrow{\sqsupseteq} EKAW : Conference\_Proceedings$

Table 1. Debugging results.

Mapping	Bridge rules count	Entailed bridge rules count	Bridge rules reduction rate	Mapping	Bridge rules count	Entailed bridge rules count	Bridge rules reduction rate
CMT-CONFTOOL*	45	34	76%	EKAW-CMT	115	96	83%
CMT-CRS*	52	38	73%	EKAW-SIGKDD	127	95	75%
CMT-SIGKDD	59	45	76%	PCS-CONFTOOL*	40	25	63%
CMT-EKAW*	111	94	85%	PCS-CRS	38	21	55%
CONFTOOL-CMT	48	34	71%	PCS-SIGKDD	56	36	64%
CONFTOOL-CRS*	65	40	62%	PCS-CMT	73	58	79%
CONFTOOL-SIGKDD	75	43	57%	PCS-EKAW*	115	96	83%
CONFTOOL-PCS	45	27	60%	SIGKDD-CMT*	59	45	76%
CRS-CMT*	50	34	68%	SIGKDD-CONFTOOL*	69	41	59%
CRS-CONFTOOL*	50	37	74%	SIGKDD-CRS*	56	34	61%
CRS-SIGKDD	57	34	60%	SIGKDD-PCS	56	36	64%
CRS-PCS	38	21	55%	SIGKDD-EKAW*	122	94	77%

Table 2. Minimization results (starred mappings were first repaired applying the debugging).

applying the heuristic debugging approach proposed in Section 3 are quite reassuring – all of the mappings automatically removed by our method are actually incorrect ones.

To estimate minimization rate we measured the initial number of bridge rules and the amount of logically entailed bridge rules discovered by applying the minimization technique. As summarized in Table 2, the amount of the entailed bridge rules in a certain automatically generated mapping varies from 50 to 80% to the initial number of bridge rules in this mapping.

5 Discussion

We have presented a method for automatically improving the result of heuristic matching systems using logical reasoning. The basic idea is similar to existing work on debugging ontologies and uses some non-standard inference methods for reasoning about mappings introduced in previous work. The method feeds on the fact that most existing matching algorithms ignore the logical implications of new mappings. This gap is filled by our method that detects malicious impacts of generated mappings and traces them back to their source. As we have shown in the experiments, in almost all cases (in fact in all cases observed in the experiment) the unwanted effects were caused by wrong mappings and we were able to remove them automatically thus improving the correctness of the generated mapping. Actually, the idea of using logical reasoning in the matching process is not new and has been proposed by others (e.g., [7, 8]), the way it is used in our work, however, is unique, as it is the only approach that takes the effects of mappings into account. We believe that this additional step can significantly improve the quality of matching methods and should be integrated in existing matching algorithms as far as they are concerned with expressive ontologies that support consistency checking. In fact, the expressiveness of the language used to encode the ontologies to be matched seems to be the only limitation of our approach which can only be applied if the language supports consistency checking. In our experiments, we have seen that we can improve the correctness of matching results by removing wrong mappings. So far, we did not quantify this improvement, this has to be done in future work.

Acknowledgements

This work was partially supported by the German Science Foundation in the Emmy-Noether Program and by the European Union under grant FP6-507482 (KnowledgeWeb) as part of the T-Rex exchange program.

References

1. A. Borgida and L. Serafini. Distributed description logics: Assimilating information from peer sources. *Journal of Data Semantics*, 1:153–184, 2003.
2. P. Bouquet, J. Euzenat, E. Franconi, L. Serafini, G. Stamou, and S. Tessaris. Specification of a common framework for characterizing alignment. Deliver. 2.2.4, KnowledgeWeb, 2004.
3. P. Bouquet, F. Giunchiglia, F. van Harmelen, L. Serafini, and H. Stuckenschmidt. C-OWL: Contextualizing ontologies. In *Proceedings of the 2nd International Semantic Web Conference (ISWC-03)*, volume 2870 of *LNCS*, pages 164–179. Springer, 2003.
4. P. Bouquet, L. Serafini, and S. Zanobini. Semantic coordination: a new approach and an application. In *Proceedings of the Second Internatinal Semantic Web Conference*, volume 2870 of *Lecture Notes in Computer Science*, pages 130–145. Springer Verlag, 2003.
5. D. Calvanese, G. De Giacomo, and M. Lenzerini. A framework for ontology integration. In *Proceedings of the Semantic Web Working Symposium*, pages 303–316, Stanford, CA, 2001.
6. M. Ehrig and S. Staab. QOM - Quick Ontology Mapping. In *Proceedings of the 3rd International Semantic Web Conference (ISWC-04)*, 2004.
7. F. Giunchiglia, P. Shvaiko, and M. Yatskevich. S-match: an algorithm and an implementation of semantic matching. In *Proceedings of the European Semantic Web Conference (ESWS-04)*, pages 61–75, 2004.
8. F. Giunchiglia, M. Yatskevich, and E. Giunchiglia. Efficient semantic matching. In *Proceedings of the European Semantic Web Conference (ESWS-05)*, pages 272–289, 2005.
9. E. Hovy. Combining and standardizing largescale, practical ontologies for machine translation and other uses. In *Proceedings of the 1st International Conference on Language Resources and Evaluation (LREC)*, pages 535–542, 1998.
10. J. Madhavan, P. A. Bernstein, P. Domingos, and A. Halevy. Representing and reasoning about mappings between domain models. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI-02)*, 2002.
11. S. Melnik, H. Garcia-Molina, and E. Rahm. Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In *Proceedings of the 18th International Conference on Data Engineering (ICDE-02)*. IEEE Computing Society, 2002.
12. N. F. Noy and M. A. Musen. The PROMPT suite: Interactive tools for ontology merging and mapping. *International Journal of Human-Computer Studies*, 59(6):983–1024, 2003.
13. L. Serafini, A. Borgida, and A. Tamilin. Aspects of distributed and modular ontology reasoning. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05)*, 2005.
14. L. Serafini, H. Stuckenschmidt, and H. Wache. A formal investigation of mapping languages for terminological knowledge. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05)*, 2005.
15. L. Serafini and A. Tamilin. DRAGO: Distributed reasoning architecture for the semantic web. In *Proceedings of the 2nd European Semantic Web Conference (ESWC-05)*, 2005.
16. H. Stuckenschmidt, H. Wache, and L. Serafini. Reasoning about ontology mappings. In *Proceedings of the ECAI-06 Workshop on Contextual Representation and Reasoning*, 2006.
17. O. Svab, V. Svatek, P. Berka, D. Rak, and P. Tomasek. Ontofarm: Towards an experimental collection of parallel ontologies. In *Poster Proceedings of the International Semantic Web Conference 2005 (ISWC-05)*, 2005.