Applying Logical Constraints to Ontology Matching

C. Meilicke, H. Stuckenschmidt

Computer Science Institute
University of Mannheim
A5, 6 68159 Mannheim, Germany
{christian, heiner} @informatik.uni-mannheim.de

Abstract. Automatically discovering semantic relations between ontologies is an important task with respect to overcoming semantic heterogeneity on the semantic web. Ontology matching systems, however, often produce erroneous mappings. In this paper we propose a method for optimizing precision and recall of existing matching systems. The principle of this method is based on the idea that it is possible to infer logical constraints by comparing subsumption relations between concepts of the ontologies to be matched. In order to verify this principle we implemented a system that uses our method as basis for optimizing mappings. We generated a set of synthetic ontologies and corresponding defective mappings and studied the behavior of our method with respect to the properties of the matching problem. The results show that our strategy actually improves the quality of the generated mappings.

1 Motivation

Recently, a number of heuristic methods for matching concepts from different ontologies have been proposed. These methods rely mostly on computing similarities based on linguistic and structural criteria. Evaluation studies have shown that existing methods often trade off precision and recall. The resulting mapping either contains a fair amount of errors or only covers a small part of the ontologies involved [2, 4]. Our goal is to provide a component for matching systems that optimizes the results with respect to both recall and precision of the generated mapping. The method that we suggest is based on a reasoning approach that goes beyond existing structural methods and can be classified as semantic-based technique due to Euzenat [3].

1.1 Problem Statement

In accordance to Euzenat [3] the problem of ontology matching can be defined in the following way. For each ontology $\mathcal{T}$ there is a function $Q(\mathcal{T})$ that defines matchable elements of $\mathcal{T}$. Given two ontologies $\mathcal{T}$ and $\mathcal{T}'$ the task of matching is to determine correspondences between the matchable elements in the two ontologies. Correspondences can be defined as 4-tuples $\langle e, e', r, c\rangle$ where $e \in Q(\mathcal{T})$, $e' \in Q'(\mathcal{T}')$, r is a semantic relation and c is a confidence value from a suitable structure $\langle D, \leq\rangle$. In this work, we only consider the simple case where e and e' are concepts and $r =' \equiv'$ but there is some evidence that our approach can also be extended to $r \in \{\sqsubseteq, \sqsupseteq, \equiv\}$.

The problem we address in this paper is the following. Given a set of correspondences M' between two ontologies $\mathcal{T}$ and $\mathcal{T}'$ generated as intermediary or final result of a matching system and a set of correspondences M that contains all correct semantic relations between elements from the two ontologies, to determine $M' \cap M$. In the following we refer to a set of correspondences between two ontologies as a mapping.

1.2 Approach and Contributions

The approach taken in this work is to interpret the problem defined above on the one hand as optimization problem with respect to the confidences of the correspondences in M'. On the other hand we assume that any acceptable solution to the problem has to fulfill additional logical constraints imposed by the logical theories encoded in the ontologies. In particular, semantic relations between ontologies should not cause any inconsistencies. Therefore, we will develop a method that enables us to find a subset of M' that is the optimal mapping among the consistent mappings in the powerset of M'. The concrete contributions of this work are:

- We propose a method for automatically optimizing automatically generated mappings based on a reasoning approach that is orthogonal to existing matching approaches.
- We give a detailed explanation of the developed algorithm and implemented it in a tool for mapping optimization.
- We applied this tool on several synthetic data sets, to identify success factors for optimizing ontology mappings in terms of properties of the mapping and the matched ontologies.

The paper is organized as follows. First, we discuss related work and explain why our approach goes beyond existing structural methods used in ontology matching. In section 2 we introduce the main principle of our approach and define the mapping property of consistency. In section 3 we describe the algorithms and components that our method is based on and show how these components can be integrated into a system for optimizing mappings. The experiments we conducted are described in section 4. In particular, we analyse the relation between certain properties of the matching problem and their influence on the results of the suggested method. We close with a general discussion of the approach and possible extensions in future work in section 5.

1.3 Related Work

A variety of methods for computing mappings have been proposed. A common feature of these methods is that they are based on an initial mapping created by matching labels that occur in the ontologies and successively improve this initial mapping by combining and updating mappings based on certain heuristics [3]. The work described in this paper is a new approach for combining and updating an initial mapping set.

So far two types of methods for improving an initial mapping have been proposed. The first kind of approaches are so-called structural techniques. They are based on the propagation of evidences for certain mapping hypotheses based on the structure of the ontologies. The GLUE system [1] uses relaxation labeling to update the probability that a mapping is correct. The main idea is that the label (concept in the target ontology) of a node (concept in the source ontology) is influenced by the features of the nodes neighborhood in the subsumption graph. The OMEN tool [11] uses a Bayesian network, where a node stands for a correspondence between classes or properties of the ontologies and where an edge represents the influences between individual correspondences. In order to generate conditional probability tables for the given network a set of meta-rules is used.

The other kind of methods are so-called semantic methods that try to infer additional correspondences or try to eliminate incorrect correspondences using logical reasoning. The idea is to encode the semantics of concepts as well as the initial mapping in a logical theory. Additional correspondences are inferred by proving implications between formulas that represent concepts in the different ontologies. In [6] a semantic matching approach is proposed that uses propositional logic for encoding the semantics of concept labels and uses a SAT prover to derive mappings. In [8] the approach is extended to the task of matching structured representations by coding them into description logics and inferring subsumption relations across ontologies. A prequel of the approach described in this paper has been suggested in [9] and [10] where we use conflict sets and distributed reasoning to eliminate potentially incorrect correspondences. Notice that the strategy suggested in [10] has only been applied in the context of mapping repairing. The algorithm to solve this problem consists of a sequence of local decisions not taking into account the whole distribution of confidence values as well as the complex interdependences between inconsistencies.

Both of these approaches have shortcomings. While structural approaches that rely on numerical methods have problems in capturing hard semantic constraints, semantic approaches that solely rely on logical reasoning are often too strict to capture all valid mappings and suffer from problems in modelling the soft constraints implied by confidence values. In our work we combine numerical and logical methods thereby leveraging the problems of the individual approaches.

2 General approach

In this section we revert to some examples based on the scenario that has been described by Quine as radical translation [14]. Radical translation is concerned with the problem of finding a correct translation manual for a fully unknown language L. Obviously, this problem is closely related to the problem of finding a correct mapping between ontologies. Therefore, it is useful to explain the intuition that forms the basis of our strategy first in the context of radical translation. Later in this section, we will shift to the problem of ontology matching and give a formal representation.

2.1 Translation

Suppose that a linguist wants to explore the unknown language L of some people that have not been in contact to human civilization yet. The native people accept the researcher and let him be part of their daily life. At the first stage of his project the linguist simply observes the linguistic behavior of the natives and establishes some hypothesis about the meaning of the words that are uttered by the natives. The following could be a typical example for such a situation.

Example 1. The linguist and a native are standing in front of an oak tree. A rabbit is sitting close to the tree. The native points at the direction of the tree and utters the word ”Gavagai!”. The linguist considers two possible hypothesis about the meaning of the word. Gavagai could on the one hand refer to oak or could on the other hand refer to rabbit. He writes both hypothesis in his dictionary and marks them with a q as questionable.

As time goes by, the linguist is able to utter simple sentences in L. He also finds out which words and gestures mean approval and rejection. After a while he also manages to ask questions of the form ”Are all x y?” translated to L. This enables him to apply a more elaborative strategy.

Example 2. From time to time the linguist cleans up the entries in his dictionary. He finds, amongst others, the following three entries.

$$gavagai = rabbit_q \tag{1}$$
$$gavagai = oak_q \tag{2}$$
$$snok = tree \tag{3}$$

In order to find out if the first or the second entry has to be removed he asks the native the question ”Are all gavagais snoks?”. The native looks quite confused and denies the question. For that reason the linguist removes the second entry and keeps the first one.

The reasoning that is the base for the linguists decision follows this line. If $gavagai$ means oak and $snok$ means $tree$ then everything that is a $gavagai$ also has to be a $snok$, because the linguist knows that an oak is a special kind of a tree. He transfers this subsumption relation to the concepts $gavagai$ and $snok$. By asking the question ”Are all gavagais snoks?” the linguist checks if this entailment is accepted by the native. The native denies this question and therefore the linguist is justified in removing the second or the third entry. Since he has marked the second entry as questionable he decides to remove it instead of removing the third entry.

2.2 Formalization

The problem of radical translation is structurally closely related to the problem of automated generation of mappings between ontologies. An ontology can be understood as a formal representation of (parts of) a language that can be used to talk about certain parts of the world. Thus, every entry in the dictionary of the linguist can be interpreted as a correspondence in an ontology mapping.

How can the strategy of the linguist be formalized in order to apply it to the problem of matching ontologies? The linguist uses his dictionary to connect both views of the world. By doing this he derives knowledge about the subsumption relations of the natives concepts. The same can be done in the context of ontologies by defining the union of two ontologies connected by a mapping in the following straight forward way.

Definition 1 (Union of ontologies). *Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be ontologies (finite sets of axioms). The union $\mathcal{T}_1 \cup_{\mathcal{M}} \mathcal{T}_2$ of $\mathcal{T}_1$ and $\mathcal{T}_2$ connected by $\mathcal{M}$ is defined as $\mathcal{T}_1 \cup_{\mathcal{M}} \mathcal{T}_2 = \mathcal{T}_1 \cup \mathcal{T}_2 \cup \{t(x) \mid x \in \mathcal{M}\}$ with t being a translation function that converts correspondences into axioms in the following way:*

$$t(\langle 1\colon C, 2\colon D, \equiv, c\rangle) \;=\; 1\colon C \equiv 2\colon D$$

Such a union ontology defines a taxonomy determined by the axioms of both ontologies and the additional correspondence axioms. Consider again example 2. The linguist first had the union ontology in mind that results from the mapping consisting of dictionary entries 2 and 3. Therefore, the linguist inferred the axiom $L: gavagai \sqsubseteq L\colon snok$. If the native would accept this axiom his ontology would become inconsistent. Thus, the native denies the question of the linguist. We call the corresponding notion mapping consistency and introduce it formally as follows.

Definition 2 (Consistency of a Mapping). *Given ontologies $\mathcal{T}_1$ and $\mathcal{T}_2$ and a mapping $\mathcal{M}$ between $\mathcal{T}_1$ and $\mathcal{T}_2$. $\mathcal{M}$ is consistent iff there exists no concept $i\colon C$ with $i \in \{1, 2\}$ such that $\mathcal{T}_i \not\models i\colon C \sqsubseteq \bot$ and $\mathcal{T}_1 \cup_{\mathcal{M}} \mathcal{T}_2 \models i\colon C \sqsubseteq \bot$. Otherwise $\mathcal{M}$ is inconsistent.*

As we have argued, an inconsistent mapping is a mapping that contains erroneous correspondences. In the next section we will define the closely related notion of pairwise mapping consistency and we will see how to apply it to the problem of mapping optimization.

3 Algorithms

The problem stated in section 1.1 can now be addressed in the following way. On the one hand we have to find a subset M^* of M' that is an approximation of the correct mapping M taking into account the confidence values of the correspondences in M'. Since we are only focussed on equivalence correspondences, we can restrict M^* to be a one-to-one mapping. A one-to-one mapping is a mapping that contains no pairs of correspondences $\langle c_1, c_2\rangle$ with $c_1 \neq c_2$ such that the source (target) concept of c_1 is also the source (target) concept of c_2. On the other hand M^* has to be a consistent mapping. Thus, we need an efficient algorithm that finds a consistent mapping M^* that is also optimal with respect to the confidence values of M'. We will now introduce such an algorithm that is based on three components. The first two components provide methods for optimization respectively checking consistency, while the third component combines these methods to solve the problem.

3.1 Optimization

Finding an optimal solution to the one-to-one matching problem based on the confidence values of the elements of M' depends on choosing an appropriate aggregation function. We decided to maximize the sum of all confidence values. The optimization problem can thus be stated in the following way:

Definition 3 (Optimal solution). *Given a set of correspondences M', an optimal one-to-one mapping $M_{opt} \subseteq M'$ is a one-to-one mapping such that for every other one-to-one mapping $M'' \subseteq M'$ we have $\sum_{c \in M_{opt}} confidence(c) \geq \sum_{c \in M''} confidence(c)$.*

A standard algorithm to solve this problem is known as the hungarian method [7]. In order to show how this method can be applied to our problem a few explanations have to be given. The hungarian method expects a real-valued matrix as input and creates a one-to-one assignment, such that the sum of the chosen entries is minimal. To use the hungarian method the input mapping M' has to be transformed into a corresponding matrix H. Each concept of the source ontology corresponds to a row and each target concept corresponds to a column. Since the hungarian method finds a minimal assignment an entry in the matrix has to be interpreted as distance between two concepts, where the distance between $1\colon C$ and $2\colon D$ is defined as $1 - confidence(\langle 1\colon C, 2\colon D, \equiv, c\rangle)$. Without loss of generality we assume that the input confidence values are in the interval $[0,1]$. If there exists no such correspondence in M' the distance is set to ∞.

In most matching situations it will not be possible to match all or even the majority of concepts. Matching candidates will thus not be available. Therefore, the input matrix has to be extended by additional concepts that play the role of alternative matching candidates. We call these concepts phantom concepts. Thus, if n is the number of concepts in $\mathcal{T}_1$ and m is the number of concepts in $\mathcal{T}_2$, we add m rows to the input matrix corresponding to m phantom concepts $1\colon P_1, \ldots, 1\colon P_m$ as well as n columns corresponding to n phantom concepts $2\colon P_1, \ldots, 2\colon P_n$. The value of the entries in these rows respectively columns is set to $1 + \epsilon$ with $\epsilon > 0$. Thus, for a given concept $1\colon C$ the algorithm will first try to find a corresponding concept $2\colon D$. If this is not possible within the context of global minimization one of the phantom concepts $2\colon P_1, \ldots, 2\colon P_n$ is chosen. In such a case $1\colon C$ is interpreted as unmatchable.

Example 3. Assume mapping M consists of the following correspondences between $\mathcal{T}_1$ and $\mathcal{T}_2$.

$$\langle 1\colon C, 2\colon X, =, 0.94\rangle \tag{4}$$

$$\langle 1\colon C, 2\colon Y, =, 0.29\rangle \tag{5}$$

$$\langle 1\colon D, 2\colon X, =, 0.12\rangle \tag{6}$$

$$\langle 1\colon E, 2\colon X, =, 0.31\rangle \tag{7}$$

The corresponding input matrix to the hungarian method looks like this:

	$j : X$	$j : Y$	$2 : P_1$	$2 : P_2$	$2 : P_3$
$1 : C$	0.06	0.71	$1+\epsilon$	$1+\epsilon$	$1+\epsilon$
$1 : D$	0.88	∞	$1+\epsilon$	$1+\epsilon$	$1+\epsilon$
$1 : E$	0.69	∞	$1+\epsilon$	$1+\epsilon$	$1+\epsilon$
$1 : P_1$	$1+\epsilon$	$1+\epsilon$	$1+\epsilon$	$1+\epsilon$	$1+\epsilon$
$1 : P_2$	$1+\epsilon$	$1+\epsilon$	$1+\epsilon$	$1+\epsilon$	$1+\epsilon$

Suppose we set $\epsilon = 9$. Applying the hungarian method will result in the one-to-one mapping consisting of correspondences 5 and 7. Concept $1\colon D$ is mapped on a phantom concept. The aggregated distance value for the chosen correspondences is 31.4. ϵ is an important parameter that affects the behaviour of the mapping extraction. Consider again example 3 with ϵ set to 0.001. The hungarian method will now find another optimal extraction consisting of only one correspondence, namely correspondence 4. This mapping extraction has an aggregated cost of 4.1. The algorithm chooses this mapping, since the cost of assinging an additional concept to a phantom concept is less than the relatively high cost of choosing two correspondences with a high distance. We will later see how to set the value of ϵ according to the properties of the matching problem to be solved.

3.2 Reasoning

The second component of our system consists of an incomplete but efficient reasoning method. This method can be used to detect mapping inconsistencies and subsets of the mapping that cause these inconsistencies. As our approach extensively relies on checking consistency in the mapped ontologies, using existing methods for reasoning about ontologies is infeasible.

Therefore, we decided to use the following method that allows us to check mapping consistency for each mapping of cardinality two. First, we use a reasoner to compute the whole concept hierarchy of both $\mathcal{T}_1$ and $\mathcal{T}_2$. For each ontology $\mathcal{T}_i$ with $i \in \{1,2\}$ we save the results in a subsumption matrix that contains the information if $i\colon C$ is a subclass of $i\colon D$ for each pair of concepts $\langle i\colon C, i\colon D\rangle$. In the same way we prepare a disjointness matrix that contains the information if two concepts are disjoint. Having once computed these four matrices the reasoning method can be implemented by comparing entries in matrices. Given a set of correspondences $M = \{\langle 1\colon C, 2\colon D, =, v_1\rangle, \langle 1\colon E, 2\colon F, =, v_2\rangle\}$ we first check if $1\colon C$ is a subclass or superclass of $1\colon E$. If this is the case $2\colon D$ and $2\colon F$ cannot be disjoint because this would result in an inconsistency in the union ontology $\mathcal{T}_1 \cup_M \mathcal{T}_2$. We also apply the same procedure in the other direction by entailing subsumption relations from $\mathcal{T}_2$ to $\mathcal{T}_1$ accompanied by checking disjointness in $\mathcal{T}_1$. Notice that inconsistencies can also emerge, if a concept $j\colon D$ becomes a subclass of $j\colon C$ even if $j\colon C$ and $j : D$ are not defined as disjoint in $\mathcal{T}_j$. This may happen if a subclass of $j\colon D$ is defined as disjoint to $j\colon C$. Therefore, all subclasses of a class that becomes subsumed have to be checked for disjointness, too.

In order to apply this strategy to a mapping M that consists of more than two correspondences we check consistency of each dual-element subset of M. This results in a correct but incomplete reasoning method for checking consistency of the whole mapping M. We call a mapping M that is consistent for all pairs of correspondences pairwise-consistent. The according property can be defined as follows.

Definition 4 (Pairwise-Consistency of a Mapping). *Given ontologies $\mathcal{T}_1$ and $\mathcal{T}_2$ and a mapping $\mathcal{M}$ between $\mathcal{T}_1$ and $\mathcal{T}_2$. $\mathcal{M}$ is pairwise-consistent iff there exists no $\mathcal{M}' \subseteq \mathcal{M}$ with $|\mathcal{M}'| = 2$ such that $\mathcal{M}'$ is inconsistent. Otherwise $\mathcal{M}$ is pairwise-inconsistent.*

An advantage of this reasoning approach, in addition to its efficiency, is the fact that we can restrict the reason for inconsistency to a pair of correspondences. Due to our considerations in the previous sections, we thus know that one of the elements in the pair cannot be accepted. Note that exactly the same kind of reasoning has been used by the linguist in example 2.

Even though most of all inconsistencies can be detected by this strategy, there are inconsistences consisting of more than two correspondences. Consider the following example.

Example 4. Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be ontologies describing the domain of conferences in a slightly different way. Suppose that both ontologies contain the concepts $Paper$ and $Poster$. In $\mathcal{T}_1$ there is a super concept $Submission$ that is defined as the union of both concepts, while in $\mathcal{T}_2$ the equivalent concept is called $Document$. $\mathcal{T}_1$ contains the additional concept $ReviewedSumission$ which is defined to be a sub concept of $Submission$. In $\mathcal{T}_2$ the concept $SubmissionDeadline$ is defined to be disjoint with $Document$. Now let $\mathcal{M}$ be a mapping between $\mathcal{T}_1$ and $\mathcal{T}_2$ that consists of the following three correspondences.

$$\langle 1\colon Paper\ , 2\colon Paper, =, 1.0\rangle \tag{8}$$

$$\langle 1\colon Poster\ , 2\colon Poster, =, 1.0\rangle \tag{9}$$

$$\langle 1\colon ReviewedSubmission\ , 2\colon SubmissionDeadline =, 0.52\rangle \tag{10}$$

By using 8 and 9 it can be inferred that $1\colon Submisson$ and $2\colon Document$ are equivalent concepts. Now we have the situation that in $\mathcal{T}_1$ $ReviewedSubmission$ is a subconcept of $Submission$ while in $\mathcal{T}_2$ $SubmissionDeadline$ is disjoint with $Document$ which results in a conflict with correspondence 10. Thus, we have an example for a mapping that is obviously inconsistent due to definition 2, but contains no inconsistent pair of correspondences due to definition 4.

Notice that example 4 is quite complex. Combinations of correspondences that follow a similar pattern will occur relatively infrequently in automatically generated mappings. In experiments on real-world ontologies this consideration could be verified.

3.3 Search

The two components described in the sections above can be combined in a uniform cost search. Thus, it is possible to find the best solution to the matching problem among the set of all pairwise-consistent solutions. The algorithm starts with the optimal solution that results from applying the hungarian method to the original matching problem as root node. Therefore, we convert the input mapping M to a distance matrix H as described in section 3.1 and compute the optimal one-to-one mapping M'. If M' is a consistent mapping, we have found a solution to the problem in the first step. If M' is not pairwise-consistent, there exists at least one conflicting pair of correspondences $\langle c_1, c_2\rangle$ in M'. We now know that a consistent solution must not contain both c_1 and c_2, while it is possible that one of these correspondences is contained in the final solution that we are searching for. Therefore, we have to consider two branches in our search tree. In the first branch we have to search for a solution that must not contain c_1, while in the second branch we have to search for a solution that must not contain c_2.

Now remember that the hungarian method will never chose a mapping that consists of a correspondence with a distance value of ∞. We can make use of this property to create the branches in the search tree. Therefore, we have to set the cell corresponding to c_1 to ∞ in the first branch while setting the cell corresponding to c_2 to ∞ in the second branch. In the following we describe this procedure as locking a cell in H, which corresponds to making the associated correspondence unavailable in this branch. In this way we create new search states based on the results of the consistency checks applied to every state that gets expanded. For each new search state we compute its aggregated minimum value by applying the hungarian method and save the state and its associated minimum in minimum-priority queue. Notice that a search state is fully described by a set of locked cells and the associated minimum with respect to a particular input matrix H. In each step of the search we expand the state with lowest minimum. Thus, our algorithm finally results in a uniform cost search that uses the techniques described in the sections above to create new search states and to decide which states to expand first.

The algorithm terminates if it expands a state that corresponds to a pairwise consistent mapping. Due to the property of the uniform cost search, to first expand a set of locks with lowest aggregated distance, any other set of locks with a lower aggregated distance must have been expanded in an earlier step of the search, and thus has to be a pairwise-inconsistent mapping. Therefore, the algorithm will always find an optimal pairwise-consistent solution to the matching problem.

The runtime of the algorithm is exponential in worst case. The actual runtime depends on the structure of the input matrix. The most influential parameters are the size of the input mapping M, the size of the ontologies, the number of pairwise-inconsistencies caused by M, the quality of the confidence ordering in M with respect to correct and incorrect correspondences, and the number of matching alternatives in M.

4 Experiments

In the following we describe some of the experiments we conducted on synthetic data sets. In these experiments we examine the relation between certain parameters of the matching problem on the one hand and precision and recall of our algorithm on the other hand. In particular, we will address the following questions:

– How does the **fraction of correct correspondences** in the mapping affect the results of our approach?
– In how far does the **fraction of concepts to be covered via correspondences** influence the results?
– How strong and under which circumstances does the **quality of the confidences** affect the optimization process?
– Does the **structure of the ontologies** influence the results of our approach?

By using synthetic datasets we can vary these parameters with respect to the question under discussion. Besides the properties of the matching problems we will also vary ϵ and study interrelations between different ϵ-values and the characteristics of the matching problem.

4.1 Experimental Settings

Data sets We construct some synthetic ontologies $\mathcal{T}_{b,d}$ where b denotes the branching factor and d the depths of the subsumption hierarchy. Further, we define sibling concepts to be disjoint. We consider one-to-one mappings $M_{b,d}$ to the same ontology, matching each concept to itself. In the experiments we randomly choose subsets of $M_{b,d}$ of varying size to represent correct correspondences between concepts. We refer to the size of this subset as coverage in the following. In addition, we add a certain amount of incorrect correspondences by linking randomly chosen concepts $C \neq D$ from $\mathcal{T}_{b,d}$. To simulate matching systems that differ in the quality of confidence estimations we assign confidence values to correct and incorrect correspondences according to certain patterns that will be explained below. Notice that a synthetic mapping M' constructed in this way has to be understood as an intermediate result of a matching system. The final result has to be extracted from M'. This final step is often done using the hungarian method [7] which produces an global optimum (see [3]). Therefore, we will always compare the synthetic mapping after applying the hungarian method, to the mapping after applying our algorithm.

Experiments The first experiment is based on the synthetic ontology $\mathcal{T}_{3,3}$ and the associated mapping $M_{3,3}$. We generated $M_{3,3}$ and decided to analyze subsets with a coverage from 0.1 to 1.0 increasing the coverage stepwise by 0.05. For each of these mappings we added n incorrect correspondences respectively $n/3$ incorrect correspondences where n is the number of concepts in the particular mapping. Thus, we created mappings with a precision of 0.5 respectively 0.75. For each mapping we distributed confidence values randomly not distinguishing between correct and incorrect correspondences. Notice that such a distribution is a challenge for every approach that tries to improve the quality of an input mapping. It can be adopted that matching systems will generate more reliable confidence estimations in most cases. Normally, the recall of a mapping M' is defined as $|M' \cap M|/|M|$, where M is the reference mapping that contains all correct correspondences between the ontologies to be matched. Since M is not known to us we compare the number of correct correspondences in the synthetic mapping to the number of correct correspondences left after applying the standard extraction respectively our algorithm and interpret the fraction as recall.

In the second experiment we are concerned with the probability of a correct correspondence having a greater confidence value than an incorrect correspondence. We refer to this probability as the perfection of a mapping. Notice that this parameter is different from the precision of a mapping. More precisely, the perfection of a mapping is the probability that a randomly chosen correct correspondence has a higher confidence than a randomly chosen incorrect correspondence. Again, we created $M_{3,3}$ and randomly chose subsets with a coverage of 0.2. In contrast to the first experiment we added more incorrect correspondences resulting in a relatively low precision of 0.3. To study different distributions of confidence values we increased perfection from 0.5 to 0.9 stepwise by 0.1. A mapping with low precision but relatively high perfection can be regarded as the intermediary result of a matching system which is optimized for recall. In order to compare the effects of different ϵ-values we applied the algorithm with $\epsilon = 0.001$ and $\epsilon = 100$. For each configuration we repeated the experiment 100 times focussing on the resulting mean values.

The third experiment deals with the issue of different subsumption hierarchies. Therefore, we decided to apply our algorithm on ontologies that differ in depth and branching factor to obtain information about the influence on the optimization process.

4.2 Experimental Results

The results of the first experiment are presented in figure 1. The left side shows the results with respect to precision, the right side refers to recall. There are four curves plotted in both parts of the figure. The data series resulting from an input mapping with a precision of 0.75 are marked with a rectangle. The data series resulting from an input mapping with a precision of 0.5 are marked with a circle. For both experiments we compared the mapping after applying the hungarian method (dashed grey line) to the mapping after applying our algorithm (black solid line). The results are based on randomly creating 500 mappings for each setting computing the mean of precision and recall with ϵ set to a high value, that forces our algorithm to find a consistent mapping that has a maximum number of elements.

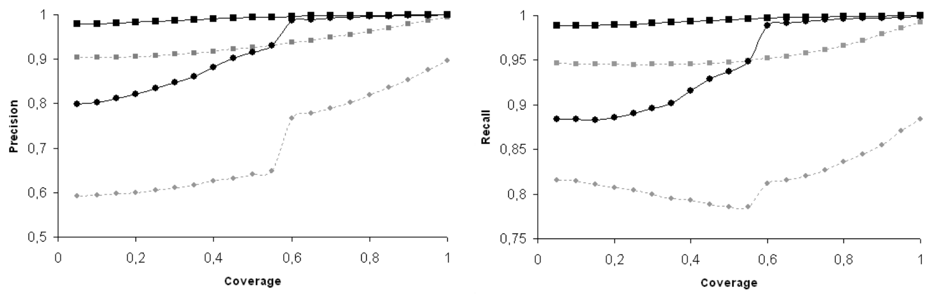

Fig. 1. Precision and recall with respect to coverage and precision of the input mapping

First, let us consider the results with respect to precision. For an input precision of 0.5 and a coverage from 0.05 to 0.6 we observe a difference of approximately 20% comparing the straight forward one-to-one mapping extraction to the results based on applying our algorithm. The differences become smaller with increasing coverage. Notice that in most real world matching problems there will only be a relatively low coverage. Even if two ontologies describe largely overlapping domains their concepts can be overlapping without being equivalent. Equivalence correspondences will thus only cover a small number of concepts. Therefore, the results for a high coverage are of minor interest. For an input precision of 0.75 we can observe a similar pattern. The precision of the one-to-one mapping extraction starts at 0.9 while applying logical constraints increases this value to 0.97 even for a coverage of 0.05. For a coverage above 0.35 the average precision of the optimized mapping is continuous greater than 0.99.

The results for recall are similar to the results for precision. As mentioned above we defined recall in our setting with respect to the correct correspondences of the input mapping. Thus, the first extraction as well as the final extraction decreases the actual recall of the input mapping that we interpreted as the initial or intermediary result of a matching system. But since any matching system that aims at generating a one-to-one mapping has to find an extraction from an intermediary result - which could e.g. be a whole similarity matrix - we are well-founded in comparing the extraction based on the hungarian method to the application of our algorithm. For settings with a low coverage we can increase recall by approximately 5% for both mappings with a precision of 0.5 and 0.75. These results look unusual at first sight. One might have expected a trade-off between precision and recall. The reason for increasing both precision and recall is based on the fact that our method does not filter out particular correspondences as long as an alternative is available. More precisely, instead of selecting a subset of the first extraction, the algorithm forces rearrangements upon the assignments available. These rearrangements will with some probability result in choosing more correct correspondences compared to the first or some of the previous mapping extractions computed in the search procedure. This approach is opposed to the method suggested in [9] where inconsistencies are used to filter out correspondences by removing the correspondence with the lowest confidence in a conflict set.

We can therefore conclude that the proposed approach is capable of increasing precision as well as recall of an input mapping to a substantial extent for both low and high input precision. Actually, our algorithm has stronger effects on input mappings with a low precision. The explanation for this observation is simply based on the fact that with increasing number of incorrect correspondences the probability of conflicting pairs of correspondences increases, too. Nevertheless, this consideration can only be extended to a certain degree. For an input mapping M with extremly low precision there could also be a subset M^* of M consisting mostly of incorrect correspondences that are pairwise-consistent. If this subset is larger than the subset of correct correspondences M' (extended by some incorrect and not conflicting correspondences) the algorithm will choose M^* instead of M'. You should also notice that there is a substantial variance not depicted in the mean values of figure 1. By taking a look at the negative outliers that cause this variance, the pattern just mentioned can be detected. Since the probability of consistent sets of type M^* decreases with increasing size of M, the variance decreases also with increasing coverage.

	$\epsilon = 0.001$			$\epsilon = 100$			
Perfection	Precision	Recall	f-value	Precision	Recall	f-value	Δ f-value
0.5	0.370	0.551	0.443	0.399	0.634	0.490	-0.047
0.6	0.389	0.589	0.468	0.393	0.630	0.484	-0.016
0.7	0.431	0.659	0.521	0.417	0.671	0.514	0.007
0.8	0.458	0.705	0.555	0.434	0.702	0.536	0.019
0.9	0.478	0.739	0.581	0.446	0.724	0.552	0.029

Table 1. The influence of perfection on precision and recall for low and high ϵ.

In the second experiment we focussed on the relation between the value of ϵ and the perfection of the input mapping. The results of this experiment are summarized in table 1. Each row in the table shows the results for a particular input perfection. Besides precision and recall we also added the f-value which is the weighted harmonic mean of both. This value provides us with an overall estimation of the performance of our algorithm. In order to compare the results for $\epsilon = 0.001$ and $\epsilon = 100$ we listed the differences of both approaches with respect to the resulting f-values in the last column.

First, we see that there is a strong positive correlation between perfection of the input mapping and precision respectively recall of the outcome. This result is not suprising. Nevertheless, it shows that the algorithm makes use of the additional information encoded in the confidence values. It is more interesting that for $\epsilon = 0.001$ this information has much stronger positive effects. With respect to the f-value we gain an advancement of 0.138 for $\epsilon = 0.001$ if we compare both extremes in perfection, while the advancement for $\epsilon = 100$ is limited to 0.062. What is the reason for this difference? Assume there are (amongst others) two overlapping pairs of conflicting correspondences $\langle c_1, c_2 \rangle$ and $\langle c_1, c_3 \rangle$ in the input mapping. There are two possibilities to solve this problem: Discard c_1 or discard c_2 and c_3. In the context of the algorithm we would put a lock on the associated cells in the matrix. If we now choose a high ϵ value the algorithm is forced to take the first option, not at all taking into account the confidence values involved. The situation changes if we set $\epsilon = 0.001$. Now the algorithm will make its decision based on comparing $confidence(c_1)+1.01$ to $confidence(c_2)+confidence(c_3)$. A decision that is based on this comparison, obviously, makes sense only if the confidence values under consideration are with some probability correct estimations. By definition the perfection of a mapping is a parameter that is characteristic for this probability. For a high perfection decisions based on the consideration explained above will expand the search tree in the correct direction with a high probability, while for a low precision these considerations will be misleading. This explains the values presented in the last column. For a low precision the conservative strategy - using a high ϵ value and therefore keeping as much correspondences as possible - works better, while for a high perfection the conservative strategy cannot exploit the additional information encoded in the confidence values to its full extent.

In the third experiment we studied the behavior of the algorithm working with ontologies that differ in their hierarchical structure. Obviously, the applicability of our algorithm relies on the existence of pairwise inconsistencies. With respect to the synthetic ontologies we used in our experiments the number of subsumption statements and disjointness statements is determined by the branching factor and the depth. Thus, we varied these parameters to understand their impact on the results. In one of our testcases we compared mappings for $\mathcal{T}_{7,2}$ (relatively flat subsumption hierarchy with 56 concepts) and $\mathcal{T}_{2,5}$ (deep subsumption hierarchy with 62 concepts). We could achieve stronger effects in optimizing mappings linking concepts of $\mathcal{T}_{2,5}$. Compared to $\mathcal{T}_{7,2}$ we measured 8% more precision and 3% more recall. There are several reasons for this difference. With respect to $\mathcal{T}_{7,2}$ pairwise inconsistencies will on the one hand only occur, if there are some correspondences in the input mapping that link concepts of the first level. On the other hand incorrect correspondences linking sibling concepts that are leaves will never cause pairwise-inconsistencies in a one-to-one mapping. If we compare results for e.g. $\mathcal{T}_{4,3}$ to $\mathcal{T}_{2,5}$ the differences become noticeable smaller. We can conclude that our approach works slightly better on matching ontologies with a deep subsumption hierarchy.

5 Discussion and Conclusions

We presented an approach to optimize matching systems based on a combination of numerical optimization and logical reasoning, thereby leveraging the problems of existing approaches that are solely based on optimization or logical reasoning, respectively.

We introduced the basic principle of the approach based on the idea of radical translation and transferred it to the problem of matching ontologies. We defined the properties of mapping consistency and pairwise mapping consistency as a basis for automating this principle. We presented an algorithm for computing an optimal and pairwise-consistent mapping that combines the hungarian method with a uniform cost search over the space of pairwise consistent mappings. We ran several experiments on synthetic data sets and showed that our method increases precision and recall of a mapping in comparison to the result of applying standard optimization techniques without considering mapping consistency. In particular, we showed that even for mappings with poor confidence estimations our approach works quite well.

There are two main lines of future work. The first is concerned with improving the efficiency of our method to scale up to large real world ontologies. While the incomplete reasoning method we use is rather efficient due to the pre-compilation of the subsumption and the disjointness matrix, the complexity of the current approach is mainly influenced by the optimization and the search procedure. The hungarian algorithm currently used for the optimization has a complexity of $O(n^3)$. Sacrificing global optimality we can improve this by replacing the hungarian method with the Gale-Shapley algorithm [5] which runs in $O(n^2)$. We expect major improvements from using more sophisticated search procedures instead of uniform cost search to deal with the combinatorial explosion of space of consistent mappings. We will also investigate the use of efficient but incomplete search methods such as greedy or stochastic local search.

The second major point for future work is the systematic application of the method to real data sets. In this paper, our aim was to better understand the behavior of our method in terms of the influence of different problem characteristics. For this purpose, artificial data sets are better suited than real ones. Now that we gained an understanding of the success factors, we are ready to tackle real matching problems. First experiments we carried out on data sets from the ontology alignment evaluation initiative have shown that the critical point here is the assumption we make about the presence of disjointness statements in the ontologies. It turned out that ontologies often do not contain such statements even though the concepts are clearly disjoint. A possible solution is to simply add disjoint statements for all sibling concepts. In [12] it has been shown that this radical approach works well in many cases. A more sophisticated approach is to try to learn disjointness statements for underspecified ontologies based on suitable text corpora and background knowledge. In [13] first results for this approach are reported that suggest that learning is indeed feasible. We will investigate and contrast these different approaches on real data in future work.

6 Acknowledgement

The work has been partially supported by the German Science Foundation (DFG) in the Emmy Noether Programme under contract STU 266/3-1.

References

1. AnHai Doan, Jayant Madhavan, Robin Dhamankar, Pedro Domingos, and Alon Halevy. Learning to match ontologies on the semantic web. *The International Journal on Very Large Data Bases*, 12:303–319, 2003.
2. Jerome Euzenat, Malgorzata Mochol, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Svab, Vojtech Svatek, Willem Robert van Hage, and Mikalai Yatskevich. First results of the ontology alignment evaluation initiative 2006. In Richard Benjamins, Jerome Euzenat, Natasha Noy, Pavel Shvaiko, Heiner Stuckenschmidt, and Michael Uschold, editors, *Proceedings of the ISWC 2006 Workshop on Ontology Matching*, Athens, GA, USA, November 2006 2006.
3. Jerome Euzenat and Pavel Shvaiko. *Ontology Matching*. Springer Verlag, 2007. To appear.
4. Jerome Euzenat, Heiner Stuckenschmidt, and Mikalai Yatskevich. Introduction to the ontology alignment evaluation 2005. In *Proceedings of the K-CAP 2005 Workshop on Integrating Ontologies*, Banff, Canada, 2005.
5. D. Gale and L. S. Shapley. College admissions and the stability of marriage. *American Mathematical Monthly*, pages 9–14, 1962.
6. F. Giunchiglia, M. Yatskevich, and P. Shvaiko. Semantic matching: Algorithms and implementation. *Journal on Data Semantics*, 2007. To appear.
7. H.W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics*, pages 83–97, 1955.
8. Simone Sceffer Paolo Bouquet Luciano Serafini, Stefano Zanobini. Matching hierarchical classifications with attributes. In *Proceedings of the 3rd European Semantic Web Conference (ESWC 2006)*, pages 4–18, 2006.
9. Christian Meilicke, Heiner Stuckenschmidt, and Andrei Tamilin. Improving automatically created mappings using logical reasoning. In *ISWC-06 Workshop on Ontology Matching*, Athens, GA, USA, 2006.
10. Christian Meilicke, Heiner Stuckenschmidt, and Andrei Tamilin. Repairing ontology mappings. In *Proceedings of the Twenty-Second Conference on Artificial Intelligence (AAAI-07)*, Vancouver, Canada, 2007.
11. Praesenjit Mitra, Natalya F. Noy, and Anuj R. Jaiswal. Omen: A probabilistic ontology mapping tool. In *ISWC-04 Workshop on Meaning Coordination and Negotiation*, Hiroshima, Japan, 2004.
12. S. Schlobach. Debugging and semantic clarification by pinpointing. In *Proceedings of ESWC 2005*, 2005.
13. Johanna Völker, Denny Vrandecic, York Sure, and Andreas Hotho. Learning disjointness. In *Proceedings of the 4th European Semantic Web Conference (ESWC'07)*. Springer, 2007.
14. W.V.Quine. *Word and Object*. MIT Press, 1960.