Repairing the missing is-a structure of ontologies

Patrick Lambrix, Qiang Liu, and He Tan

Department of Computer and Information Science
Linköpings universitet
581 83 Linköping, Sweden

Abstract. Developing ontologies is not an easy task and often the resulting ontologies are not consistent or complete. Such ontologies, although often useful, also lead to problems when used in semantically-enabled applications. Wrong conclusions may be derived or valid conclusions may be missed. To deal with this problem we may want to repair the ontologies. Up to date most work has been performed on finding and repairing the semantic defects such as unsatisfiable concepts and inconsistent ontologies. In this paper we tackle the problem of repairing modeling defects and in particular, the repairing of structural relations (is-a hierarchy) in the ontologies. We study the case where missing is-a relations are given. We define the notion of a structural repair and develop algorithms to compute repairing actions that would allow deriving the missing is-a relations in the repaired ontology. Further, we define preferences between repairs. We also look at how we can use external knowledge to recommend repairing actions to a domain expert. Further, we discuss an implemented prototype and its use as well as an experiment using the ontologies of the Anatomy track of the Ontology Alignment Evaluation Initiative.

This is a preprint of the paper published by Springer-Verlag:
Lambrix P, Liu Q, Tan H, Repairing the missing is-a structure of ontologies, Proceedings of the 4th Asian Semantic Web Conference - ASWC09, LNCS 5926, 76-90, Shanghai, China, 2009. *The original publication is available at www.springerlink.com. doi: 10.1007/978-3-642-10871-6_6*

1 Introduction

Developing ontologies is not an easy task and often the resulting ontologies are not consistent or complete. Such ontologies, although often useful, also lead to problems when used in semantically-enabled applications. Wrong conclusions may be derived or valid conclusions may be missed. Defects in ontologies can take different forms (e.g. [7]). Syntactic defects are usually easy to find and to resolve. Defects regarding style include such things as unintended redundancy. More interesting and severe defects are the modeling defects which require domain knowledge to detect and resolve, and semantic defects such as unsatisfiable concepts and inconsistent ontologies. Most work up to date has focused on finding and repairing the semantic defects in an ontology (e.g. [13, 7, 6, 5]). Recent work has also started looking at repairing semantic defects in a set of mapped ontologies [4] or the mappings between ontologies themselves [11].

In this paper we tackle the other difficult problem, i.e. the repairing modeling defects. In particular, we focus on the repairing of structural relations (is-a hierarchy) in the ontologies. In this setting it is known that a number of intended is-a relations are not present in the source ontology. The missing is-a relations can be discovered by inspection of the ontologies by experts or they can be generated by automated tools. For instance, in the case of task 4 in the Anatomy track in the 2008 Ontology Alignment Evaluation Initiative (OAEI) [10], two ontologies, Adult Mouse Anatomy Dictionary [1] (MA, 2744 concepts) and the NCI Thesaurus - anatomy [12] (NCI-A, 3304 concepts), and 988 mappings between the two ontologies are given. Based on the structure of the source ontologies and the given mappings, it can be derived that 178 is-a relations in MA and 146 in NCI-A are missing.[1]

Once missing is-a relations are found, the problem is to add is-a relations (or subsumption axioms) to the ontology such that the missing is-a relations can be derived. Although the easiest way to do this, is to just add the missing is-a relations, this may not be the most interesting solution for the domain expert. For instance, in MA an is-a relation between wrist joint and joint is missing and could be added to the ontology. However, knowing that there is an is-a relation between wrist joint and limb joint in MA, a domain expert may want to add an is-a relation between limb joint and joint. This is more informative and would lead to the fact that the missing is-a relation can be derived. In general, such a decision is preferably made by a domain expert. Therefore, in this work, we develop algorithms to generate and recommend possible ways to repair the structure of the ontology and develop a tool that allows a domain expert to repair the structure of an ontology in a semi-automatic way.

In section 2 we formally define the notion of structural repair. As not all possible ways to repair an ontology are equally useful, we also define a number of preference relations between repairs. Section 3 describes our algorithms for generating, recommending and executing repairing actions. Our prototype system and its use are described in section 4. Further, we discuss experiments on repairing MA and NCI-A in section 5. Related work is presented in section 6 and the paper concludes in section 7.

2 Theory

The setting that we study is the case where the ontology is defined using named concepts and subsumption axioms[2]. Most ontologies contain this case and many of the most well-known and used ontologies, e.g. in the life sciences, are covered by this setting. We therefore use the following definition.

Definition 1. *Let $\mathcal{O} = (\mathcal{C}, \mathcal{I})$ be an ontology with $\mathcal{C}$ its set of named concepts and $\mathcal{I} \subseteq \mathcal{C} \times \mathcal{C}$ a representation of its is-a structure. Let $\mathcal{M} \subseteq \mathcal{C} \times \mathcal{C}$ be a set of missing is-a relations (i.e. $\mathcal{M}$ represents a set of missing subsumptiom axioms). A* **structural repair** *for the ontology $\mathcal{O}$ with respect to $\mathcal{M}$ is a set of pairs of concepts $\mathcal{R} \subseteq \mathcal{C} \times \mathcal{C}$ such that for each $(A_i, B_i) \in \mathcal{M}$: $(\mathcal{C}, \mathcal{I} \cup \mathcal{R}) \models A_i \rightarrow B_i$.*

[1] A number of these are actually redundant. For instance, it may be that when repairing one missing is-a relation, others are repaired as well. Using this property we can remove 57 missing is-a relations from MA (with 121 remaining) and 63 from NCI-A (with 83 remaining).

[2] In this paper we denote subsumption axioms often using $\rightarrow$. A $\rightarrow$ B means that A is-a B.

The definition states that a structural repair of an ontology given a set of missing is-a relations, is a set of is-a relations such that when these is-a relations are added to the ontology, then all missing is-a relations can be derived from the extended ontology. The elements in a structural repair we call *repairing actions*.

An immediate consequence is that the set of missing is-a relations is itself a structural repair. Another consequence is that adding is-a relations to a structural repair also constitutes a structural repair.

Not all structural repairs are equally useful or interesting for a domain expert. To deal with this issue we introduce a number of preference relations.

In some structural repairs there may be is-a relations that do not contribute to the derivation of the missing is-a relations. In other structural repairs some of the is-a relations may be derivable from the other is-a relations in the structural repair and therefore redundant. For example, the missing is-a relation between wrist joint and joint can be repaired in MA by adding the is-a relation between limb joint and joint. In that case, the missing is-a relation between elbow joint and joint is also repaired since there is a is-a relation between elbow joint and limb joint. Therefore, an is-a relation between elbow joint and joint in the structural repair is redundant. The first preference relation prefers not to use these redundant or non-contributing is-a relations for repairing.

Definition 2. *Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be structural repairs for the ontology $\mathcal{O}$ with respect to $\mathcal{M}$, then $\mathcal{R}_1$ is axiom-preferred to $\mathcal{R}_2$ (notation $\mathcal{R}_1 \ll_A \mathcal{R}_2$) iff $\mathcal{R}_1 \subseteq \mathcal{R}_2$.*

As discussed in the introduction, just adding the missing is-a relations, is not always the most interesting solution for the domain expert. For instance, repairing the missing is-a relation wrist joint and joint by adding an is-a relation between limb joint and joint may be more informative. When one is-a relation can be derived from another in the context of the ontology, we say that the second is-a relation is more informative than the first. The second preference relation prefers to use as informative is-a relations as possible for repairing.

Definition 3. *We say that (X_1,Y_1) is more informative than (X_2,Y_2) iff $X_2 \rightarrow X_1$ and $Y_1 \rightarrow Y_2$. Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be structural repairs for the ontology $\mathcal{O}$ with respect to $\mathcal{M}$. Then $\mathcal{R}_1$ is information-preferred to $\mathcal{R}_2$ (notation $\mathcal{R}_1 \ll_I \mathcal{R}_2$) iff $\exists$ (X_1,Y_1) $\in \mathcal{R}_1$, (X_2,Y_2) $\in \mathcal{R}_2$: (X_1,Y_1) is more informative than (X_2,Y_2).*

Further, some structural repairs may introduce equivalence relations for concepts which were only connected by an is-a relation in the original ontology. Although such a structural repair may result in a consistent ontology, this is usually not desired from a modeling perspective. For example, in MA we have is-a relations between posterior communicating artery and artery, and between communicating artery and artery. However, there is a missing is-a relation between posterior communicating artery and communicating artery. This could be repaired by adding an is-a relation between artery and communicating artery. However, this also introduces an equivalence between communicating artery and artery. The third preference relation prefers not to change is-a relations in the original ontology into equivalence relations.

Definition 4. *Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be structural repairs for the ontology $\mathcal{O}$ = ($\mathcal{C}$, $\mathcal{I}$) with respect to $\mathcal{M}$. Then $\mathcal{R}_1$ is strict-hierarchy-preferred to $\mathcal{R}_2$ (notation $\mathcal{R}_1 \ll_{SH} \mathcal{R}_2$) iff*

Input:
Source ontology, missing is-a relations.
Output
Repairing actions.
Algorithm
1. Initialize KB with ontology;
2. For every missing is-a relation (A_i,B_i): add the axiom $A_i \rightarrow B_i$ to the KB;
3. For each (A_i, B_i):
    3.1 Source(A_i, B_i) := super-concepts(A_i) - super-concepts(B_i);
    3.2 Target(A_i, B_i) := sub-concepts(B_i) - sub-concepts(A_i);
4. Missing is-a relation (A_i, B_i) can be repaired by choosing an element
    from Source(A_i, B_i) x Target(A_i, B_i).

Fig. 1. Algorithm for generating repairing actions - 1.

$\exists$ *A, B* $\in \mathcal{C}$*: (*$\mathcal{C}$*,* $\mathcal{I}$*)* $\models$ *A* $\rightarrow$ *B and (*$\mathcal{C}$*,* $\mathcal{I}$*)* $\not\models$ *B* $\rightarrow$ *A and (*$\mathcal{C}$*,* $\mathcal{I} \cup \mathcal{R}_1$*)* $\not\models$ *B* $\rightarrow$ *A and (*$\mathcal{C}$*,* $\mathcal{I} \cup \mathcal{R}_2$*)* $\models$ *B* $\rightarrow$ *A.*

In general, we would want structural repairs that are maximally preferred.

Definition 5. *A structural repair* $\mathcal{R}$ *for the ontology* $\mathcal{O}$ *with respect to* $\mathcal{M}$ *is maximally preferred with respect to the preference relation* $\ll$ *iff for all structural repairs* $\mathcal{R}_1$ *for* $\mathcal{O}$ *with respect to* $\mathcal{M}$ *it holds that if* $\mathcal{R}_1 \ll \mathcal{R}$ *then* $\mathcal{R} \ll \mathcal{R}_1$.

3 Repairing the structure of an ontology

A naive way to compute all possible structural repairs would be to take all sub-sets of $\mathcal{C}$ x $\mathcal{C}$ and for each sub-set, add its elements as is-a relations to the ontology and check whether the missing is-a relations can be derived. This is in practice infeasible as it requires checking too many cases. Even for small ontologies, it is not practical as domain experts usually deal with one or a few missing is-a relations at a time, rather than choosing between large sets of possible repairs including all missing is-a relations. Therefore, we develop algorithms that generate possible repairing actions for the missing is-a relations, taking into account the preferences defined in section 2. We also provide an algorithm that recommends repairing actions. The user can then select a missing is-a relation to repair (and we rank these in terms of the number of possible repairing actions). Further, we developed an algorithm that, upon the repairing of a missing is-a relation, detects for which missing is-a relations the set of repairing actions needs to be updated, and updates these.

3.1 Generating repairing actions

Algorithm 1 In our first algorithm (see figure 1), when generating repairing actions for a missing is-a relation, we take into consideration that all missing is-a relations will be repaired, but we do not take into account the actual repairing actions that could be performed for other missing is-a relations.

In the algorithm we store the ontology in a knowledge base and add the missing is-a relations to the ontology. As we know that these should be derivable in the repaired ontology, adding them introduces the desired new connections. Then, we generate correct ways to introduce more informative is-a relations that would allow us to derive the missing is-a relations. Therefore, for a repairing action (S_i,T_i) regarding missing is-a relation (A_i, B_i) we require that $A_i \rightarrow S_i$ and $T_i \rightarrow B_i$ (preference $\ll_I$ in definition 3). This also ensures that we only compute repairing actions that are relevant for repairing the missing is-a relations (preference $\ll_A$ in definition 2.) At the same time we do not want to introduce new equivalence relations, where in the source ontology we have only is-a relations (preference $\ll_{SH}$ in definition 4). This is realized by the selection of the elements in the Source and Target sets.

The proposed repairing actions for a missing is-a relation (A_i, B_i) all lead to the derivation of (A_i, B_i) in the extended ontology. In general, a user may repair the ontology by choosing for each missing is-a relation (A_i, B_i) an element from Source(A_i, B_i) and an element from Target(A_i, B_i). However, as we have not taken into account all influences of possible repairing actions for other missing is-a relations, a better strategy is to repair one missing is-a relation and recompute repairing actions for the other missing is-a relations in the partially repaired ontology.

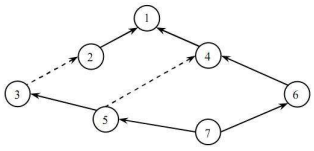

Fig. 2. Example 1.

As an example, consider the case presented in figure 2, where $\mathcal{O}_1$ = ($\mathcal{C}_1$, $\mathcal{I}_1$) is an ontology with concepts $\mathcal{C}_1$ = {1, 2, 3, 4, 5, 6, 7} and is-a relations (shown in full lines in figure 2) $\mathcal{I}_1$ = {(7,5), (7,6), (5,3), (2,1), (6,4), (4,1)}. ($\mathcal{I}_1$ represents the is-a hierarchy and thus also all is-a relations derived from the elements in $\mathcal{I}_1$.) The set of missing is-a relations (shown in dashed lines in figure 2) is $\mathcal{M}_1$ = {(5,4), (3,2)}. The algorithm will then generate the following Source and Target sets: Source(5,4) = {5, 3, 2, 1, 4} - {4, 1} = {5, 3, 2}; Target(5,4) = {4, 6, 7, 5} - {5, 7} = {4, 6}; Source(3,2) = {3, 2, 1} - {2, 1} = {3}; Target(3,2) = {2, 3, 5, 7} - {3, 5, 7} = {2}. For missing is-a relation (3,2) the only generated repairing action is (3,2). For missing is-a relation (5,4) any of the repairing actions (5,4), (5,6), (3,4), (3,6), (2,4), (2,6) together with (any of) the generated repairing action(s) for (3,2) leads to the derivation of the missing is-a relation (5,4) in the extended ontology. The example also shows the importance of initially adding the missing is-a relations to the knowledge base. The possible repairing action (2,4) for missing is-a relation (5,4) would not be generated when we do not take

Input:
Source ontology, missing is-a relations.
Output
Repairing actions.
Algorithm
1. Initialize KB with ontology;
2. For every missing is-a relation $A_i \rightarrow B_i$:
    2.1 create new concepts X_i and Y_i in the KB;
    2.2 add the axioms $A_i \rightarrow X_i$, $X_i \rightarrow Y_i$, $Y_i \rightarrow B_i$ to the KB;
3. For each (A_i, B_i):
    3.1 Source-ext(A_i, B_i) := super-concepts(A_i) - super-concepts(X_i);
    3.2 Target-ext(A_i, B_i) := sub-concepts(B_i) - sub-concepts(Y_i);
4. Missing is-a relation (A_i, B_i) can be repaired by choosing an original ontology element from Source-ext(A_i, B_i) and an original ontology element from Target-ext(A_i, B_i).

Fig. 3. Algorithm for generating repairing actions - 2.

into account that missing is-a relation (3,2) will be repaired.[3] Further, the example also shows that we do not introduce repairing actions that would turn is-a relations in the original ontology into equivalence relations. For instance, adding (1,4) would lead to the fact that missing is-a relation (5,4) would be derivable in the extended ontology, but also leads to making 1 and 4 equivalent.

Algorithm 2 Our second algorithm for finding repairing actions for a particular missing is-a relation (see figure 3) takes into account influences of other missing is-a relations that are valid for all possible choices for repairing actions for the other missing is-a relations. The difference between the basic algorithm and our extended algorithm occurs mainly in steps 2 and 3. Instead of adding the missing is-a relations to the knowledge base, in the extended algorithm we introduce for each missing is-a relation (A_i, B_i) two new concepts X_i and Y_i in the knowledge base as well as the axioms $A_i \rightarrow X_i$, $X_i \rightarrow Y_i$, $Y_i \rightarrow B_i$. (X_i, Y_i) satisfies the requirements that each possible repairing action for (A_i, B_i) should satisfy. As they are new concepts in the knowledge base, the properties and relations of X_i, respectively Y_i, to other concepts in the knowledge base represent the properties and relations that are common to the source concepts, respectively target concepts, of the possible repairing actions for (A_i, B_i). The Source and Target sets are now computed relative to the X_i and Y_i.

As an example, consider the case presented in figure 4, where $\mathcal{O}_2 = (\mathcal{C}_2, \mathcal{I}_2)$ is an ontology with concepts $\mathcal{C}_2 = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ and is-a relations (shown in full lines in figure 4) $\mathcal{I}_2 = \{(7,6), (6,5), (5,2), (2,1), (7,4), (10,4), (10,9), (9,8), (8,3), (3,1), (4,1)\}$. (As before, $\mathcal{I}_2$ represents the is-a hierarchy and thus also all is-a relations derived from the elements in $\mathcal{I}_2$.) The set of missing is-a relations (shown in dashed

[3] So this means that repairing one is-a relation may influence the repairing actions for other missing is-a relations. However, when *generating* repairing actions in algorithm 1 the only influence that is taken into consideration is the fact that missing is-a relations are or will be repaired (least informative repairing action), but not the actual (possibly more informative) repairing actions that could be performed.

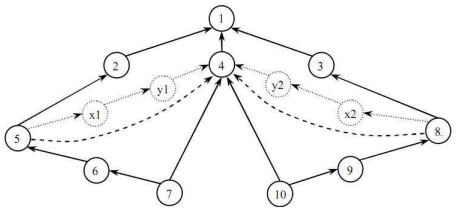

Fig. 4. Example 2.

lines in figure 4) is $\mathcal{M}_2$ = {(5,4), (8,4)}. The algorithm in figure 1 will then generate the following Source and Target sets: Source(5,4) = {5, 4, 1, 2} - {4, 1} = {5, 2}; Target(5,4) = {4, 8, 9, 10, 5, 6, 7} - {5, 6, 7} = {4, 8, 9, 10}; Source(8,4) = {8, 4, 1, 3} - {4, 1} = {8, 3}; Target(8,4) = {4, 8, 9, 10, 5, 6, 7} - {8, 9, 10} = {4, 5, 6, 7}.

The extended algorithm in figure 3 will add the nodes x1, y1, x2, y2 and the is-a relations 5 → x1, x1 → y1, y1 → 4, 8 → x2, x2 → y2, and y2 → 4 (shown in dotted lines in figure 4). It then generates the following Source and Target sets: Source-ext(5,4) = {5, 4, 1, 2, x1, y1} - {4, 1, x1, y1} = {5, 2}; Target-ext(5,4) = {4, 8, 9, 10, 5, 6, 7, x1, y1, x2, y2} - {5, 6, 7, x1, y1} = {4, 8, 9, 10, x2, y2}; Source-ext(8,4) = {8, 4, 1, 3, x2, y2} - {4, 1, x2, y2} = {8, 3}; Target-ext(8,4) = {4, 8, 9, 10, 5, 6, 7, x1, y1, x2, y2} - {8, 9, 10, x2, y2} = {4, 5, 6, 7, x1, y1}. The sets generated by the extended algorithm indicate that there is an influence between the two missing is-a relations. Indeed, when a choice is made for repairing the first missing is-a relation, we have essentially added equivalence relations between x1, respectively y1, and concepts in the ontology. The appearance of x1 and y1 in the Target-ext set for the second missing is-a relation indicates that the concept chosen to be equivalent to x1 (and all concepts between this concept and 5) are now also candidates for the Target for the second missing is-a relation. For example, when choosing (2,4) as a repairing action for missing is-a relation (5,4) then (3,2) is a possible repairing action for missing is-a relation (8,4).

Similarly to the basic algorithm, the proposed repairing actions for a missing is-a relation (A_i, B_i) all lead to the derivation of (A_i, B_i) in the extended ontology. In general, a user may repair the ontology by choosing for each missing is-a relation (A_i, B_i) an element from Source(A_i, B_i) and an element from Target(A_i, B_i). However, as the algorithm only takes into account influences that are common to all possible choices for repairing actions, a user may want to repair one missing is-a relation and recompute repairing actions for the other missing is-a relations.

3.2 Recommending repairing actions

As there may be many possible repairing actions, we develop a method for recommending repairing actions based on domain knowledge. We assume that we can query the domain knowledge regarding subsumption of concepts. There are several such sources such as general thesauri (e.g. WordNet) or specialized domain-specific sources (e.g the Unified Medical Language System). In our algorithm (see figure 5) we generate recommended repairing actions for a missing is-a relation starting from the Source and Target

Input:
domain knowledge, source ontology, missing is-a relation (A_i, B_i),
Source and Target for the missing is-a relation as computed by algorithm in figure 1.
Output
Recommended repairing actions.
Algorithm
Global Variable $visited$: stores already processed repairing actions.
Global Variable $recommended$: stores recommended repairing actions.
1. Set $visited$ = {(A_i, B_i)};
2. Set $recommended$ = {(A_i, B_i)};
3. Set X_e = {x_e : $x_e \in$ Source(A_i, B_i) $\bigwedge \forall x \in$ Source(A_i, B_i): if $x_e \rightarrow x$ then $x = x_e$};
4. Set Y_e = {y_e : $y_e \in$ Target(A_i, B_i) $\bigwedge \forall\, y \in$ Target(A_i, B_i): if $y \rightarrow y_e$ then $y = y_e$};
5. For each pair $(x_e, y_e) \in X_e$ x Y_e: call `QCheck`(x_e, y_e);
6. Return $recommended$;
Function `QCheck(concept x, concept y)`
i. If $(x, y) \in visited$ then return;
ii. Add (x, y) to $visited$;
iii. If $\exists\ (x_r, y_r) \in recommended$: $x \rightarrow x_r \wedge y_r \rightarrow y$ then return;
iv. If x is a sub-concept of y according to the domain knowledge then
    Remove all (x_r, y_r) from $recommended$ for which $x_r \rightarrow x$ and $y \rightarrow y_r$;
    add (x, y) to $recommended$;
  else
    Let Y_{sup} be the set of direct super-concepts of y;
    For each $y_s \in Y_{sup} \cap$ Target(A_i, B_i): call `QCheck`(x, y_s);
    Let X_{sub} be the set of direct sub-concepts of x;
    For each $x_s \in X_{sub} \cap$ Source(A_i, B_i): call `QCheck`(x_s, y);

Fig. 5. Algorithm for recommending repairing actions.

sets generated by the algorithm in figure 1[4]. The algorithm selects the most informative repairing actions that are supported by evidence in the domain knowledge. The variable $visited$ in the algorithm in figure 5 keeps track of already processed repairing actions. The variable $recommended$ stores recommended repairing actions at each step and its final value is returned as output. It is initialized with the missing is-a relation itself. This is the least informative repairing action that can be performed for repairing the missing is-a relation. Steps 3 and 4 compute the set X_e of maximal elements with respect to the is-a relation in the Source set and the set Y_e of minimal elements with respect to the is-a relation in the Target set. The elements from X_e x Y_e are then the most informative repairing actions. For each of these elements (x,y) we check whether there is support in the domain knowledge in step 5. Steps i and ii in the function QCheck do bookkeeping regarding the already processed repairing actions. Step iii assures that we do not add recommended is-a relations that are less informative than others already recommended. In step iv we check whether there is support in the domain knowledge for the repairing action. If so, then the repairing action is recommended and all less informative repairing

[4] We have also extended the algorithm in figure 5 to deal with Source and Target sets derived by the algorithm in figure 3.

actions are removed from the recommendation set. If not, then we check whether there is support in the domain knowledge for the repairing actions that are less informative than (x,y). Among these we start with the most informative repairing actions.

3.3 Executing repairing actions

When a user has chosen a repairing action for a particular missing is-a relation, it may influence the set of possible repairing actions for other missing is-a relations. Therefore, the repairing actions for the other missing is-a relations need to be recomputed based on the ontology extended with the chosen repairing action.

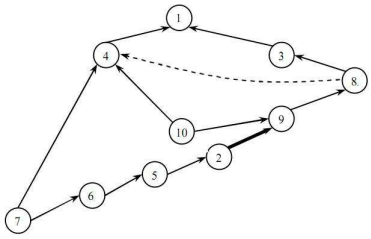

Fig. 6. Example 2 - update.

For instance, figure 6 shows the new situation when choosing the repairing action (2,9) (shown in thick line) for repairing missing is-a relation (5,4) for the example in figure 4. In this case the Source and Targets sets become the following for the basic algorithm: Source(8,4) = {8, 4, 1, 3} - {4, 1} = {8, 3}; Target(8,4) = {4, 8, 9, 10, 2, 5, 6, 7} - {8, 9, 10, 2, 5, 6, 7} = {4}; and the following for the extended algorithm: Source-ext(8,4) = {8, 4, 1, 3, x2, y2} - {4, 1, x2, y2} = {8, 3}; Target-ext(8,4) = {4, 8, 9, 10, 2, 5, 6, 7, x2, y2} - {8, 9, 10, 2, 5, 6, 7, x2, y2} = {4}. When we compare the computed repairing actions after the choice of (2,9) for repairing (5,4) with the repairing actions computed before the choice (see section 3.1), we note that the repairing actions that introduce equivalence relations (e.g. (8,6)) are removed after the choice of (2,9) (preference $\ll_{SH}$ in definition 4). However, before (2,9) is chosen these repairing actions do not necessarily introduce equivalence relations. For instance, we could have repaired (8,4) first using one of these actions, and afterwards repaired (5,4).

For small ontologies, computing the repairing actions does not take much time and the approach is feasible in a real setting. For large ontologies the computation time may not be small enough to guarantee immediate updates in an implemented tool for repairing. Therefore, in the algorithm[5] in figure 7 we have introduced a way to keep track of the influences between different missing is-a relations. The missing is-a relations for which the Source and Target sets can change are the missing is-a relations for which

[5] The algorithm in figure 7 deals with the case when we use the basic algorithm for finding repairing actions. We also have a version for when we use the extended algorithm.

Input
Ontology, the repaired missing is-a relation(A_r, B_r), the repair action (X_r, Y_r) taken for (A_r, B_r), the set of non-repaired missing relations $\mathcal{M}_r$.
Output
Updated Source and Target sets.
Algorithm
1. Add (X_r, Y_r) to the KB;
2. For each missing is-a relation $(A_i, B_i) \in \mathcal{M}_r$:
    2.1 If $A_i \rightarrow X_r$ then recompute super-concepts(A_i);
    2.2 If $B_i \rightarrow X_r$ then recompute super-concepts(B_i);
    2.3 If $A_i \rightarrow X_r$ or $B_i \rightarrow X_r$ then Source(A_i, B_i) := super-concepts(A_i) - super-concepts(B_i);
    2.4 If $Y_r \rightarrow A_i$ then recompute sub-concepts(A_i);
    2.5 If $Y_r \rightarrow B_i$ then recompute sub-concepts(B_i);
    2.6 If $Y_r \rightarrow A_i$ or $Y_r \rightarrow B_i$ then Target(A_i, B_i) := sub-concepts(B_i) - sub-concepts(A_i);

Fig. 7. Algorithm for updating repairing actions.

at least one of the concepts is a sub-concept or super-concept of at least one of the concepts in the chosen repairing action for the repaired missing is-a relation. We only update the Source and Target sets for these missing is-a relations. In addition, we also remove the other missing is-a relations that have been repaired by the current repairing action.

3.4 Ranking missing is-a relations

In general, there may be many missing is-a relations that need to be repaired. Although it is possible to repair the missing is-a relations in any order, it may be easier for the user to start with the ones where there are the fewest choices. We have therefore implemented an algorithm that ranks the missing is-a relations according to the size of the Source(A_i, B_i) x Target(A_i, B_i). The missing is-a relations with the fewest number of elements in its set are presented highest in the list of missing is-a relations.

4 Implemented system

We have implemented a prototype system that allows a user to repair the structure of an ontology using the algorithms described in section 3. We show its use using a piece of MA regarding the concept Joint. As input our system takes an ontology in OWL format as well as a list of missing is-a relations[6]. We use a framework and reasoner provided by Jena (version 2.5.7) [3]. The domain knowledge that we use is WordNet [15] and the Unified Medical Language System [14].

The ontology and missing is-a relations can be imported using the *Load/Derive Missing IS-A Relations* button. The user can see the list of missing is-a relations under

[6] We actually also allow to add two ontologies together with mappings. The system will then derive missing is-a relations for an ontology based on the other ontology and the mappings in a similar way as the approach described in [8].

Fig. 8. Missing is-a relations.

the *Missing IS-A Relations* menu (see figure 8). In this case there are 7 missing is-a relations[7]. Clicking on the *Compute Repairing Actions* button, results in the computation of the Source and Target sets and the missing is-a relations in the list are ranked as described in section 3.4. The user can select which one to repair first. The first missing is-a relation in the list has the fewest possible repairing actions, and may therefore be a good starting point. When the user chooses a missing is-a relation, the Source and Target sets for the repairing actions are shown in the panels on the left and the right, respectively. The concepts in the missing is-a relation are highlighted in red.

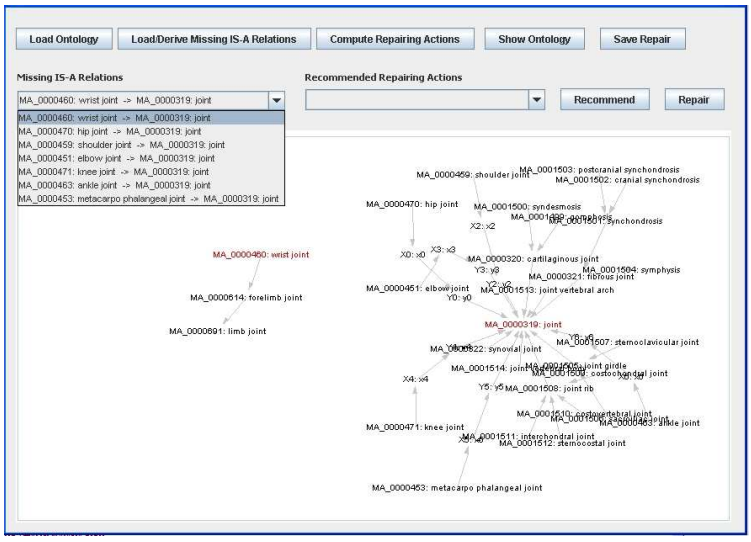

Fig. 9. Possible repairing actions for the selected missing is-a relation.

[7] The missing is-a relations were actually derived using NCI-A and mappings between MA and NCI-A.

Figure 9 illustrates the Source and Target sets for the missing is-a relation between wrist joint and joint as they were generated by our extended algorithm from figure 3. We see that, as the Target set displays x's and y's, there are a number of influences from other missing is-a relations. For instance, through x4 and y4, we see that repairing (knee joint, joint) may influence the repairing actions of the current missing is-a relation. The user can also ask for recommended repairing actions by clicking the *Recommend* button. In our case, the system recommends to add an is-a relation between limb joint and joint. In general, the system presents a list of recommendations. By selecting an element in the list, the concepts in the repairing action are highlighted in the panels. The user can repair a missing is-a relation by selecting a concept in the Source panel and a concept in the Target panel and clicking on the *Repair* button. The repairing action is then added to the ontology, and the relevant Source and Target sets and recommendations for other missing is-a relations are updated. At all times during the process the user can inspect the ontology by clicking the *Show Ontology* button. Newly added is-a relations will be highlighted (see figure 10). After adding the is-a relation between limb joint and joint, not only (wrist joint,joint) is repaired, but all other missing is-a relations as well, as they can be derived in the extended ontology. The list of missing is-a relations is therefore updated to be empty. After completing the repair of all missing is-a relations, the repaired ontology can be exported into an OWL file by clicking the *Save Repair* button.

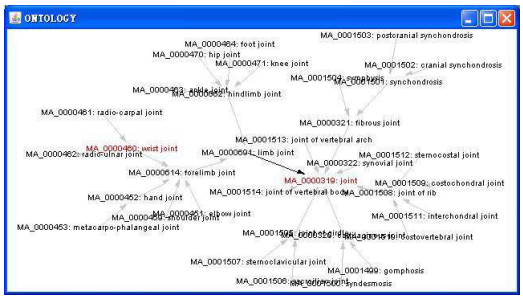

Fig. 10. The repaired ontology.

5 Experiment

In our experiment we repair the two ontologies from the 2008 Anatomy track in OAEI. As described before, MA contains 2744 concepts and NCI-A contains 3304 concepts. Using the 988 mappings between the two ontologies, it can be derived that 178 is-a relations in MA and 146 in NCI-A are missing. After removing redundancy, we still have 121 missing is-a relations for MA and 83 for NCI-A. In the remainder we use these smaller sets of missing is-a relations.

	total	1	2-10	11-20	21-30	31-40	41-50	51-100	101-200	201-300	301-400	>400
MA - Source	121	76	45	0	0	0	0	0	0	0	0	0
MA - Target	121	17	50	5	9	4	6	5	18	3	0	4
NCI-A - Source	83	28	55	0	0	0	0	0	0	0	0	0
NCI-A - Target	83	11	52	6	2	0	0	5	4	1	2	0

Fig. 11. Sizes of Source and Target sets.

	total	1	2	3	4	5	6	7	8	9	10	11-15	16-35	ST
MA	92	23	5	3	0	25	9	9	0	4	0	13	0	1
NCI-A	67	15	21	3	1	2	0	0	0	0	0	6	6	13

Fig. 12. Influence between repairing actions of different missing is-a relations - in Source or Target.

Generating repairing actions. For MA our basic algorithm generates for 15 missing is-a relations only 1 repairing action (which is then the missing is-a relation itself). This means that these could be immediately repaired. For NCI-A this number is 8. Of the remaining missing is-a relations there are 65 missing is-a relations for MA that have only 1 element in the Source and 2 missing is-relations that have 1 element in the Target set. For NCI-A these numbers are 20 and 3, respectively. These are likely to be good starting points for repairing. Figure 11 shows for different ranges how many Source and Targets sets had a size in that range. We see that for most of the missing is-a relations these sets are small and thus can be easily visualized in the panels of our system.

Figure 12 shows the influences between different missing is-a relations that can be computed using our extended algorithm. In figure 12 the last column (ST) shows the number of missing is-a relations where x's and y's of other missing is-a relations occur in both Source and Target sets. For the other columns the x's and y's only occur in Source or Target, but not in both. For instance, for MA there are 23 missing is-a relations whose Source or Target set contain x and y from one other missing is-a relation. We see that for a majority of the missing is-a relations (92/121 for MA and 67/83 for NCI-A) there are influences. An interesting observation is that in several cases missing is-a relations that have the same number of influences from other missing is-a relations, actually influence each other. For instance, in NCI-A we find missing is-a relations between each of Bronchus_Basement_Membrane, Bronchus_Cartilage, Bronchus_Lamina_Propria, Bronchus_Submucosa, and the concept Bronchus_Connective_Tissue. Repairing one of these missing is-a relations influences the repairing actions of all the others. We found several such clusters, among others for instance, in MA concerning body cavity/lining, lymphoid tissue, and brain nucleus with 7, 4 and 6 missing is-a relations, respectively.

Recommending repairing actions. In the experiment with the full ontologies we generated recommendations using WordNet only. The running time for generating recommendations for all missing is-a relations was circa 40 minutes for MA and circa

1 hour for NCI-A. In our tool, however, we do not generate recommendations for all missing is-a relations at once, but only on demand for a particular missing is-a relation.

For NCI-A the system recommended[8] repairing actions for only 5 missing is-a relations and each of those received one recommended repairing action. For MA 22 missing is-a relations received 1 recommended repairing action, 12 received 2 and 2 received 3. The recommendation can come from small sets of repairing actions or from large sets. For instance, for MA the system recommends for the missing is-a relation (mandible, bone) the three following repairing actions (oral region cartilage/bone, bone), (viscerocranium bone, bone), and (mandible, lower jaw). The repairing actions are recommended from a Source set of 177 concepts (and 15 influencing missing is-a relations) and a Target set of 3 concepts.

Executing repairing actions. To obtain information on the time it could take to repair real-case ontologies, as well as on the influences of the updates, we have run repairing sessions for MA and NCI-A with the basic algorithm. This test run was done by the authors. As we are not domain experts, we have used [2] to decide on possible choices and used the recommendation algorithm, although we cannot guarantee the correctness of our repairs. Clearly, we aim to redo this experiment with domain experts. However, this run already gives us some interesting information. After the ontologies were loaded and the first repairing actions were computed, the test run for NCI-A took about 40 minutes and for MA circa 90 minutes. In most cases the recommendations seemed useful. In the NCI-A session one missing is-a relation was removed as a result of repairing other is-a relations; in the MA session 18 were removed in three steps. Repairing influenced the number of repairing actions for other missing is-a relations. For the last 13 missing is-a relations for NCI-A (of 83 to start with) and 28 for MA (of 121 to start with) the Target set was too large to have a good visualization in the tool.

6 Related Work

We are not aware of other work that addresses the problem of repairing missing structure in ontologies. The closest is our work in [8] where we used structural repair in the context of ontology alignment. One of the methods included repairing the source ontologies by adding the missing is-a relations derived from a partial reference alignment, i.e. a set of given mappings, and the structure of the ontologies. Essentially, we used a least informative repair. However, in [8] there was no intention of trying to find better ways to repair the ontologies.

Other work that looks at the problem of repairing modeling defects is [9], where ontology repair is used when a formula can be derived from an ontology, but, in the words of the authors, it is not correct according to the world. In this case a mapping is computed such that the mapped formula is correct according to the world and can be derived from the mapped ontology, or such that the mapped formula cannot be derived from the mapped ontology. The setting where this is used is a framework where agents use ontologies and when certain tasks cannot be performed, communication between the agents takes place to identify mismatches between the ontologies and revise them.

[8] We do not count the missing is-a relation itself as a recommendation.

There is more work that addresses repairing semantic defects in ontologies. In [13] minimal sets of axioms are identified which need to be removed to turn an ontology coherent. In [7, 6, 5] strategies are described for repairing unsatisfiable concepts, explanation of errors, ranking erroneous axioms, and generating repair plans. In [4] and [11] the setting is extended to repairing mapped ontologies. In this case semantic defects may be introduced by integrating ontologies. In [4] semantic defects are repaired by removing axioms in the source ontologies, while in [11] repairing removes mappings. The solutions are often based on the computation of minimal unsatisfiability-preserving sets or minimal conflict sets.

7 Conclusion

In this paper we introduced algorithms and a tool for repairing missing is-a relations in an ontology. We defined the notion of structural repairs and developed algorithms for generating, recommending and executing repairing actions. We also discussed an experiment for repairing the two ontologies of the Anatomy track of OAEI.

There are a number of directions that are interesting for future work. In our experiment we have repaired MA and NCI-A separately. However, as we have mappings between them, we want to investigate whether repairing them together could influence the quality of the generation or recommendation of repairing actions. Further, it may also be interesting to investigate possible influences between semantic defects and modeling effects. Regarding the user interface we intend two work on at least the following issues. For large ontologies with many missing is-a relations, the first generation of repairing actions may take time and thus we want to investigate ways to partition the set of missing is-a relations into parts that can be processed independently. Further, we want to investigate new ways to visualize the Source and Target sets.

References

1. AMA. Adult mouse anatomical dictionary. http://www.informatics.jax.org/searches/AMA_form.shtml.
2. F Feneis and W Dauber. *Pocket Atlas of Human Anatomy, 4th ed.* Thieme Verlag, 2000.
3. Jena. http://jena.sourceforge.net/.
4. Q Ji, P Haase, G Qi, P Hitzler, and S Stadtmuller. RaDON - repair and diagnosis in ontology networks. In *Demo at the 6th European Semantic Web Conference*, pages 863–867, 2009.
5. A Kalyanpur. *Debugging and Repair of OWL Ontologies*. PhD thesis, University of Maryland, College Park, 2006.
6. A Kalyanpur, B Parsia, E Sirin, and B Cuenca-Grau. Repairing unsatisfiable concepts in OWL ontologies. In *Proceedings of the 3rd European Semantic Web Conference*, pages 170–184, 2006.
7. A Kalyanpur, B Parsia, E Sirin, and J Hendler. Debugging unsatisfiable classes in OWL ontologies. *Journal of Web Semantics*, 3(4):268–293, 2006.
8. P Lambrix and Q Liu. Using partial reference alignments to align ontologies. In *Proceedings of the 6th European Semantic Web Conference*, pages 188–202, 2009.
9. F McNeill and A Bundy. Dynamic, automatic, first-order ontology repair by diagnosis of failed plan execution. *International Journal on Semantic Web & Information Systems*, 3(3):1–35, 2007.

10. C Meilicke and H Stuckenschmidt. Anatomy track at the 2008 Ontology Alignment Evaluation Initiative. Anatomy at http://oaei.ontologymatching.org/2008/.
11. C Meilicke, H Stuckenschmidt, and A Tamilin. Repairing ontology mappings. In *Proceedings of the Twenty-Second National Conference on Artificial Intelligence - AAAI*, pages 1408–1413, 2007.
12. NCI-A. National cancer institute - anatomy. http://www.cancer.gov/cancerinfo/terminologyresources/.
13. S Schlobach. Debugging and semantic clarification by pinpointing. In *Proceedings of the 2nd European Semantic Web Conference*, pages 226–240, 2005.
14. UMLS. Unified medical language system. http://www.nlm.nih.gov/research/umls/about_umls.html.
15. WordNet. http://wordnet.princeton.edu/.