A Unified Approach for Aligning Taxonomies and Debugging Taxonomies and Their Alignments

Valentina Ivanova and Patrick Lambrix

Department of Computer and Information Science and the Swedish e-Science Research Centre
Linköping University, 581 83 Linköping, Sweden

Abstract. With the increased use of ontologies in semantically-enabled applications, the issues of debugging and aligning ontologies have become increasingly important. The quality of the results of such applications is directly dependent on the quality of the ontologies and mappings between the ontologies they employ. A key step towards achieving high quality ontologies and mappings is discovering and resolving modeling defects, e.g., wrong or missing relations and mappings. In this paper we present a unified framework for aligning taxonomies, the most used kind of ontologies, and debugging taxonomies and their alignments, where ontology alignment is treated as a special kind of debugging. Our framework supports the detection and repairing of missing and wrong is-a structure in taxonomies, as well as the detection and repairing of missing (alignment) and wrong mappings between ontologies. Further, we implemented a system based on this framework and demonstrate its benefits through experiments with ontologies from the Ontology Alignment Evaluation Initiative.

1 Motivation

To obtain high-quality results in semantically-enabled applications such as the ontology-based text mining and search applications, high-quality ontologies and alignments are both necessary. However, neither developing nor aligning ontologies are easy tasks, and as the ontologies grow in size, it is difficult to ensure the correctness and completeness of the structure of the ontologies. For instance, some structural relations may be missing or some existing or derivable relations may be unintended. This is not an uncommon case. It is well known that people who are not expert in knowledge representation often misuse and confuse equivalence, is-a and part-of (e.g., [2]). Further, ontology alignment systems are used for generating alignments and, as shown in the Ontology Alignment Evaluation Initiative (OAEI, http://oaei.ontologymatching.org/), alignments usually contain mistakes and are incomplete. Such ontologies and alignments, although often useful, lead to problems when used in semantically-enabled applications. Wrong conclusions may be derived or valid conclusions may be missed.

A key step towards high-quality ontologies and alignments is debugging the ontologies and alignments. During the recent years several approaches have been proposed for debugging semantic defects in ontologies, such as unsatisfiable concepts or inconsistent ontologies (e.g., [24,14,15,8]) and related to mappings (e.g., [22,11,23,28]) or integrated ontologies [13]. Further, there has been some work on detecting modeling defects (e.g., [9,3]) such as missing relations, and repairing modeling defects [19,18,16].

P. Cimiano et al. (Eds.): ESWC 2013, LNCS 7882, pp. 1–15, 2013.

The increased interest in this field has also led to the creation of an international workshop on this topic [20]. In a separate sub-field of ontology engineering, ontology alignment, the correctness and completeness of the alignments has traditionally received much attention (e.g., [25]). Systems have been developed that generate alignments and in some cases validation of alignments is supported.

In this paper we propose a unified approach for ontology debugging and ontology alignment, where ontology alignment can be seen as a special kind of debugging. We propose an integrated framework that, although it can be used as an ontology debugging framework or an ontology alignment framework, presents additional benefits for both and leads to an overall improvement of the quality of the ontologies and the alignments. The ontology alignment provides new information that can be used for debugging and the debugging provides new information that can be used by the ontology alignment. Further, the framework allows for the interleaving of different debugging and alignment phases, thereby in an iterative way continuously generating new information and improving the quality of the information used by the framework.

In sections 3, 4, 5 and 6 we propose our unified approach for ontology alignment and debugging. To our knowledge this is the first approach that integrates ontology debugging and ontology alignment in a uniform way and that allows for a strong interleaving of these tasks. We present a framework (Section 3), algorithms for the components (Sections 4 and 5) and their interactions (Section 6). Further, we show the advantages of our approach in Section 7 through experiments with the ontologies and alignment of the OAEI 2011 Anatomy track. Related work is given in Section 8 and the paper concludes in Section 9. However, we start with some preliminaries.

2 Preliminaries

In this section we introduce notions that are needed for our approach. This paper focuses on **taxonomies** $\mathcal{O} = (\mathcal{C}, \mathcal{I})$, the most widely used type of ontologies, where $\mathcal{C}$ is a set of atomic concepts and $\mathcal{I} \subseteq \mathcal{C} \times \mathcal{C}$ represents a set of atomic concept subsumptions (is-a relations). In the following we use 'ontologies' and 'taxonomies' interchangeably. An **alignment** between ontologies $\mathcal{O}_i$ and $\mathcal{O}_j$ is represented by a set $\mathcal{M}_{ij}$ of mappings between concepts in different ontologies. The concepts that participate in mappings are called **mapped concepts**. Each mapped concept can participate in multiple mappings and alignments. We currently consider equivalence mappings ($\equiv$) and is-a mappings (subsumed-by ($\rightarrow$) and subsumes ($\leftarrow$)).

The output of ontology alignment systems are **mapping suggestions**. These should be validated by a domain expert and if accepted, they become part of an alignment.

Definition 1. *A* **taxonomy network** $\mathcal{N}$ *is a tuple* $(\mathbb{O}, \mathbb{M})$ *with* $\mathbb{O} = \{\mathcal{O}_k\}_{k=1}^{n}$ *the set of the ontologies in the network and* $\mathbb{M} = \{\mathcal{M}_{ij}\}_{i,j=1;i<j}^{n}$ *the set of representations for the alignments between these ontologies.*

Figure 1 shows a small ontology network with two ontologies (concepts are represented by nodes and the is-a structures are represented by directed edges) and an alignment

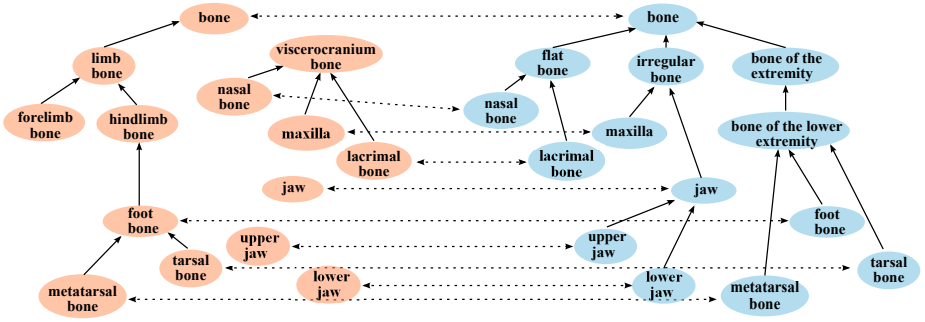

Fig. 1. (Part of an) Ontology network

(represented by dashed edges).[1] The alignment consists of 10 equivalence mappings. One of these mappings represents the fact that the concept *bone* in the first ontology is equivalent to the concept *bone* in the second ontology.

The domain knowledge inherent (logically derivable) in the network is represented by its **induced ontology**, an ontology that consists of the set of all concepts from the taxonomies, all asserted is-a relations in the taxonomies and all mappings.

In our algorithms we use **knowledge bases** (KBs) related to the taxonomies and taxonomy networks that allow us to do deductive inference.

3 Approach and Algorithms

Our framework consists of two major components - a debugging component and an alignment component. They can be used independently or in close interaction. The alignment component detects and repairs missing and wrong mappings between ontologies, while the debugging component additionally detects and repairs missing and wrong is-a structure in ontologies. Although we describe the two components separately, in our framework ontology alignment can be seen as a special kind of debugging.

The workflow (Figure 2) in both components consists of three phases during which wrong and missing is-a relations/mappings are detected (**Phase 1**), validated (**Phase 2**) and repaired (**Phase 3**) in a semi-automatic manner by a domain expert. Although the algorithms for repairing are different for missing and wrong is-a relations/mappings, the repairing goes through the same phases as shown in the figure - the generation of repairing actions (**Phase 3.1**), the ranking of is-a relations/mappings (**Phase 3.2**), the recommendation of repairing actions (**Phase 3.3**) and finally, the execution of repairing actions (**Phase 3.4**). In our approach we repair ontologies and alignments one at a time since dealing with all ontologies and alignments simultaneously would be infeasible. The is-a relations are handled in the context of the selected ontology, while the mappings are handled in the context of the selected alignment and its pair of ontologies.

[1] The first ontology is a part of AMA, the second ontology is a part of NCI-A, and the alignment is a part of the alignment between AMA and NCI-A as defined in OAEI 2011.

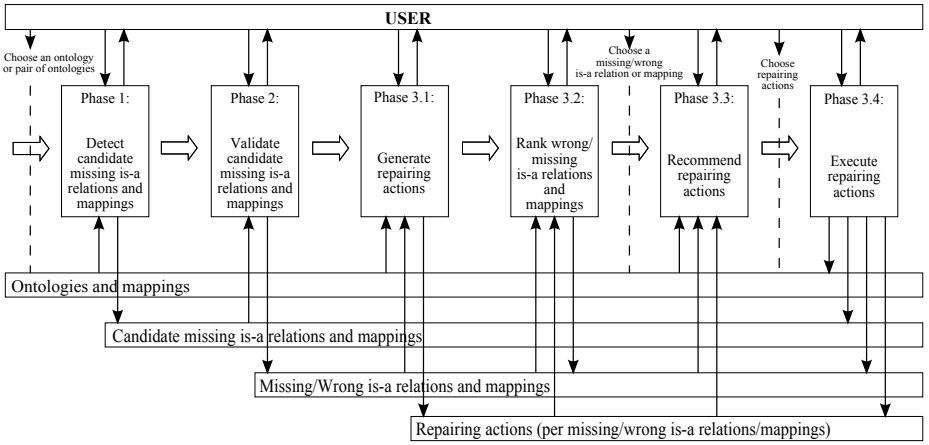

Fig. 2. Workflow

We note that at any time during the debugging/alignment workflow, the user can switch between different ontologies and the different phases shown in Figure 2. We also note that the repairing of defects often leads to the discovery of new defects, i.e., leading to additional debugging opportunities. Thus several iterations are usually needed for completing the debugging/alignment process. The process ends when no more missing or wrong is-a relations and mappings are detected or need to be repaired.

In the next three sections we describe the components and their interactions, and present algorithms for the different components and phases.

4 Debugging Component

The input for the debugging component is a taxonomy network, i.e., a set of taxonomies and their alignments. The output is the set of repaired taxonomies and alignments.

Phase 1: Detect Candidate Missing Is-a Relations and Mappings. In this component we focus on detecting wrong and missing is-a relations and mappings in the ontology network, based on knowledge that is inherent in the network. Therefore, given an ontology network, we use the domain knowledge represented by the ontology network to detect the deduced is-a relations and mappings in the network.

In our algorithm we initialize a KB for the ontology network (KB_N), KBs for each ontology (KB_k) and for each pair of ontologies and their alignment (KB_{ij}). For each ontology in the network, the set of **candidate missing is-a relations** (CMIs) derivable from the ontology network consists of is-a relations between two concepts of the ontology, which can be inferred using logical derivation from the domain knowledge inherent in the network, but not from the ontology alone. Similarly, for each pair of ontologies in the network, the set of **candidate missing mappings** (CMMs) derivable from the ontology network consists of mappings between concepts in the two ontologies, which can be inferred using logical derivation from the domain knowledge inherent in the network, but not from the two ontologies and their alignment alone.

Definition 2. *Let $\mathcal{N} = (\mathbb{O}, \mathbb{M})$ be an ontology network, with $\mathbb{O} = \{\mathcal{O}_k\}_{k=1}^{n}$, $\mathbb{M} = \{\mathcal{M}_{ij}\}_{i,j=1;i<j}^{n}$ and induced ontology $\mathcal{O}_N = (\mathcal{C}_N, \mathcal{I}_N)$. Let $\mathcal{O}_k = (\mathcal{C}_k, \mathcal{I}_k)$. Then, we define the following.*

(1) $\forall k \in 1..n$: $\mathbf{CMI}_k = \{(a,b) \in \mathcal{C}_k \times \mathcal{C}_k \mid \mathcal{O}_N \models a \rightarrow b \wedge \mathcal{O}_k \not\models a \rightarrow b\}$ *is the set of candidate missing is-a relations for $\mathcal{O}_k$ derivable from the network.*
(2) $\forall i,j \in 1..n, i < j$: $\mathbf{CMM}_{ij} = \{(a,b) \in (\mathcal{C}_i \times \mathcal{C}_j) \cup (\mathcal{C}_j \times \mathcal{C}_i) \mid \mathcal{O}_N \models a \rightarrow b \wedge (\mathcal{C}_i \cup \mathcal{C}_j, \mathcal{I}_i \cup \mathcal{I}_j \cup \mathcal{M}_{ij}) \not\models a \rightarrow b\}$ *is the set of candidate missing mappings for $(\mathcal{O}_i, \mathcal{O}_j, \mathcal{M}_{ij})$ derivable from the network.*
(3) $\mathbf{CMI} = \cup_{k=1}^{n}$ $\mathbf{CMI}_k$ *is the set of* **candidate missing is-a relations derivable from the network**.
(4) $\mathbf{CMM} = \cup_{i,j=1;i<j}^{n}$ $\mathbf{CMM}_{ij}$ *is the set of* **candidate missing mappings derivable from the network**.

In the network in Figure 1 the CMIs are *(nasal bone, bone)*, *(maxilla, bone)*, *(lacrimal bone, bone)*, *(jaw, bone)*, *(upper jaw, jaw)* and *(lower jaw, jaw)* in AMA, and *(metatarsal bone, foot bone)* and *(tarsal bone, foot bone)* in NCI-A.

Our algorithms for detecting CMIs/CMMs rely on the knowledge inherent in the network where the ontologies are connected in a network through mapped concepts. Thus the derivation paths of all CMIs and CMMs, which can be found using the knowledge inherent in the network, go through mapped concepts. Therefore, instead of checking whether the is-a relations between all pairs of concepts are derivable in the network, we only check all pairs of mapped concepts.[2,3]

Phase 2: Validate Candidate Missing Is-a Relations and Mappings. Since the structure of the ontologies may contain wrong is-a relations and the alignments may contain wrong mappings, some of the CMIs and CMMs may be derived due to some wrong is-a relations and mappings. Therefore they have to be validated by a domain expert. During Phase 2 the domain expert validates the CMIs/CMMs and partitions them into **wrong** and **missing** is-a relations/mappings. As an aid to the domain expert, we have developed recommendation algorithms based on the existence of is-a and part-of relations in the ontologies and external domain knowledge (WordNet [29] and UMLS [27]). In addition, the domain expert is provided with the derivation paths (*justifications*) for the CMI/CMM under validation.

In the network in Figure 1 *(upper jaw, jaw)* and *(lower jaw, jaw)* are validated as wrong since an upper/lower jaw is *a part-of* (not *is-a*) a jaw. The others are missing.

Phase 3: Repair Wrong and Missing Is-a Relations and Mappings. Once missing and wrong is-a relations and mappings have been obtained[4], we need to repair them. For each ontology in the network, we want to repair the is-a structure in such a way

[2] In the worst case scenario the number of mapped concept pairs is equal to the total number of concept pairs. In practice, the use of mapped concepts may significantly reduce the search space, e.g., when some ontologies are smaller than other ontologies in the network or when not all concepts participate in mappings. For instance, in the experiments in Section 7 the search space is reduced by almost 90%.

[3] For large ontologies or ontology networks, checking all pairs of concepts is also infeasible.

[4] Using the technique for detection described above or the techniques used by the alignment component or any other technique.

that (i) the missing is-a relations can be derived from their repaired host ontologies and for each pair of ontologies, we want to repair the mappings in such a way that (ii) the missing mappings can be derived from the repaired host ontologies of their mapped concepts and the repaired alignment between the host ontologies of the mapped concepts. Further (iii) the wrong is-a relations and (iv) the wrong mappings should no longer be derivable from the repaired ontology network. The notion of **structural repair** formalizes this. It contains is-a relations and mappings that should be added to or removed from the ontologies and alignments to satisfy these requirements. These is-a relations and mappings are called **repairing actions**.

Definition 3. *Let $\mathcal{N} = (\mathbb{O}, \mathbb{M})$ be an ontology network, with $\mathbb{O} = \{\mathcal{O}_k\}_{k=1}^{n}$, $\mathbb{M} = \{\mathcal{M}_{ij}\}_{i,j=1;i<j}^{n}$ and induced ontology $\mathcal{O}_N = (\mathcal{C}_N, \mathcal{I}_N)$. Let $\mathcal{O}_k = (\mathcal{C}_k, \mathcal{I}_k)$. Let $\mathcal{MI}_k$ and $\mathcal{WI}_k$ be the missing, respectively wrong, is-a relations for ontology $\mathcal{O}_k$ and let $\mathcal{MI}_N = \cup_{k=1}^{n}\mathcal{MI}_k$ and $\mathcal{WI}_N = \cup_{k=1}^{n}\mathcal{WI}_k$. Let $\mathcal{MM}_{ij}$ and $\mathcal{WM}_{ij}$ be the missing, respectively wrong, mappings between ontologies $\mathcal{O}_i$ and $\mathcal{O}_j$ and let $\mathcal{MM}_N = \cup_{i,j=1;i<j}^{n}\mathcal{MM}_{ij}$ and $\mathcal{WM}_N = \cup_{i,j=1;i<j}^{n}\mathcal{WM}_{ij}$. A* **structural repair for $\mathcal{N}$ with respect to** $(\mathcal{MI}_N, \mathcal{WI}_N, \mathcal{MM}_N, \mathcal{WM}_N)$*, denoted by $(\mathcal{R}^+, \mathcal{R}^-)$, is a pair of sets of is-a relations and mappings, such that*

(1) $\mathcal{R}^- \cap \mathcal{R}^+ = \emptyset$

(2) $\mathcal{R}^- = \mathcal{R}_M^- \cup \mathcal{R}_I^-$; $\mathcal{R}_M^- \subseteq \cup_{i,j=1,i<j}^{n}\mathcal{M}_{ij}$; $\mathcal{R}_I^- \subseteq \cup_{k=1}^{n}\mathcal{I}_k$

(3) $\mathcal{R}^+ = \mathcal{R}_M^+ \cup \mathcal{R}_I^+$; $\mathcal{R}_M^+ \subseteq \cup_{i,j=1,i<j}^{n}((\mathcal{C}_i \times \mathcal{C}_j) \setminus \mathcal{M}_{ij})$; $\mathcal{R}_I^+ \subseteq \cup_{k=1}^{n}((\mathcal{C}_k \times \mathcal{C}_k) \setminus \mathcal{I}_k)$

(4) $\forall k \in 1..n : \forall (a,b) \in \mathcal{MI}_k$: $(\mathcal{C}_k, (\mathcal{I}_k \cup (\mathcal{R}_I^+ \cap (\mathcal{C}_k \times \mathcal{C}_k))) \setminus \mathcal{R}_I^-) \models a \rightarrow b$

(5) $\forall i, j \in 1..n, i < j : \forall (a,b) \in \mathcal{MM}_{ij}$: $((\mathcal{C}_i \cup \mathcal{C}_j), (\mathcal{I}_i \cup ((\mathcal{C}_i \times \mathcal{C}_i) \cap \mathcal{R}_I^+) \cup \mathcal{I}_j \cup ((\mathcal{C}_j \times \mathcal{C}_j) \cap \mathcal{R}_I^+) \cup \mathcal{M}_{ij} \cup ((\mathcal{C}_i \times \mathcal{C}_j) \cap \mathcal{R}_M^+)) \setminus \mathcal{R}^-) \models a \rightarrow b$

(6) $\forall (a,b) \in \mathcal{WI}_N \cup \mathcal{WM}_N \cup \mathcal{R}^-$: $(\mathcal{C}_N, (\mathcal{I}_N \cup \mathcal{R}^+) \setminus \mathcal{R}^-) \not\models a \rightarrow b$

In our algorithm, at the start of the repairing phase we add all missing is-a relations and mappings to the relevant KBs. As these are validated to be correct, this is extra knowledge that should be used in the repairing process. Adding the missing is-a relations and mappings essentially means that we have repaired these using the least informative repairing actions ($\ll_I$ preference in [19]). Then during the repairing process we try to improve this and find more informative repairing actions. We say that a repairing action is more informative than another repairing action if adding the former to the ontology also allows to derive the latter. In general, more informative repairing actions that are correct according to the domain are preferred.

Definition 4. *Let (x_1, y_1) and (x_2, y_2) be two different is-a relations in the same ontology $\mathcal{O}$ (i.e., $x_1 \not\equiv x_2$ or $y_1 \not\equiv y_2$), then we say that (x_1, y_1) is* **more informative than** *(x_2, y_2) iff $\mathcal{O} \models x_2 \rightarrow x_1 \wedge y_1 \rightarrow y_2$.*

As an example, consider the missing is-a relation *(nasal bone, bone)* in Figure 1. Knowing that *nasal bone* $\rightarrow$ *viscerocranium bone*, according to the definition of more informative, we know that *(viscerocranium bone, bone)* is more informative than *(nasal bone, bone)*. As *viscerocranium bone* actually is a sub-concept of *bone* according to the domain, a domain expert would prefer to use the more informative repairing action.

Further, we initialize global variables for the current sets of missing ($\mathcal{MI}$) and wrong ($\mathcal{WI}$) is-a relations, and the current sets of missing ($\mathcal{MM}$) and wrong ($\mathcal{WM}$) mappings based on the validation results. Further, the sets of added ($\mathcal{R}_I^+$, $\mathcal{R}_M^+$) and removed

1. Compute $AllJust(w, r, \mathcal{O}_e)$
   where $\mathcal{O}_e = (\mathcal{C}_e, \mathcal{I}_e)$ such that $\mathcal{C}_e = \cup_{k=1}^{n} \mathcal{C}_k$ and
   $\mathcal{I}_e = ((\cup_{k=1}^{n} \mathcal{I}_k) \cup (\cup_{i,j=1;i<j}^{n} \mathcal{M}_{ij}) \cup \mathcal{MI}_N \cup \mathcal{MM}_N \cup \mathcal{R}_I^+ \cup \mathcal{R}_M^+) \setminus (\mathcal{R}_I^- \cup \mathcal{R}_M^-)$;
2. For every $\mathcal{I}' \in AllJust(w, r, \mathcal{O}_e)$:
   choose one element from $\mathcal{I}' \setminus (\mathcal{MI}_N \cup \mathcal{MM}_N \cup \mathcal{R}_I^+ \cup \mathcal{R}_M^+)$ to remove;

Fig. 3. Algorithm for generating repairing actions for wrong is-a relations and mappings

($\mathcal{R}_I^-$, $\mathcal{R}_M^-$) repairing actions for is-a relations and mappings, and the current sets of CMIs ($\mathcal{CMI}$) and CMMs ($\mathcal{CMM}$) are initialized to $\emptyset$.

Phase 3.1: Generate Repairing Actions. The structural repairs generated from the repairing algorithms below follow the preferences defined in [19].

Wrong Is-a Relations and Mappings. The algorithm for generating repairing actions (Figure 3) computes all justifications (AllJust) for all wrong is-a relations ($\mathcal{WI}$) and mappings ($\mathcal{WM}$). A justification for a wrong is-a relation or mapping can be seen as an explanation for why this is-a relation or mapping is derivable from the network.

Definition 5. *(similar definition as in [13]) Given an ontology $\mathcal{O} = (\mathcal{C}, \mathcal{I})$, and $(a, b) \in \mathcal{C} \times \mathcal{C}$ an is-a relation derivable from $\mathcal{O}$, then, $\mathcal{I}' \subseteq \mathcal{I}$ is a* **justification** *for (a, b) in $\mathcal{O}$, denoted by $\mathsf{Just}(\mathcal{I}', a, b, \mathcal{O})$ iff (i) $(\mathcal{C}, \mathcal{I}') \models a \rightarrow b$; and (ii) there is no $\mathcal{I}'' \subsetneq \mathcal{I}'$ such that $(\mathcal{C}, \mathcal{I}'') \models a \rightarrow b$. We use $\mathsf{All\_Just}(a, b, \mathcal{O})$ to denote the set of all justifications for (a, b) in $\mathcal{O}$.*

Our algorithm initializes a KB taking into account repairing actions up to now and computes the minimal hitting sets for each wrong is-a relation or mapping. The wrong is-a relation or mapping can then be repaired by removing at least one element in every justification.

In the network in Figure 1 *(upper jaw, jaw)* in AMA is validated as wrong. Its justification is *AMA:upper jaw* $\equiv$ *NCI-A:Upper_Jaw* $\rightarrow$ *NCI-A:Jaw* $\equiv$ *AMA:jaw*. To repair it *NCI-A:Upper_Jaw* $\rightarrow$ *NCI-A:Jaw* should be removed from NCI-A.

Missing Is-a Relations and Mappings. It was shown in [16] that repairing missing is-a relations (and mappings) can be seen as a generalized TBox abduction problem. Figure 4 shows our solution, an extension of the algorithm in [19], for the computation of repairing actions for a missing is-a relation or mapping. The main component of the algorithm (GenerateRepairingActions) computes, for a missing is-a relation or mapping, the more general concepts of the first concept (Source) and the more specific concepts of the second concept (Target) in the KB. To not introduce non-validated equivalence relations where in the original ontologies and alignments there are only is-a relations, we remove the super-concepts of the second concept from Source, and the sub-concepts of the first concept from Target. The already known wrong is-a relations or mappings and their repairing actions are removed from Repair (Source $\times$ Target). Adding an element from Repair to the KB makes the missing is-a relation or mapping derivable.

Repair missing is-a relation (a,b) with a $\in \mathcal{O}_k$ and b $\in \mathcal{O}_k$:
Choose an element from GenerateRepairingActions(a, b, KB_k);

Repair missing mapping (a,b) with a $\in \mathcal{O}_i$ and b $\in \mathcal{O}_j$:
Choose an element from GenerateRepairingActions(a, b, KB_{ij});

GenerateRepairingActions(a, b, KB):
1. $Source(a,b)$:= super-concepts(a) $-$ super-concepts(b) in KB;
2. $Target(a,b)$:= sub-concepts(b) $-$ sub-concepts(a) in KB;
3. $Repair(a,b)$:= $Source(a,b) \times Target(a,b)$;
4. For each $(s,t) \in Source(a,b) \times Target(a,b)$:
   if $(s,t) \in \mathcal{WI} \cup \mathcal{WM} \cup \mathcal{R}_I^- \cup \mathcal{R}_M^-$ then remove (s,t) from $Repair(a,b)$;
   else if $\exists (u,v) \in \mathcal{WI} \cup \mathcal{WM} \cup \mathcal{R}_I^- \cup \mathcal{R}_M^-$: (s,t) is more informative than (u,v) in KB
      and $u \rightarrow s$ and $t \rightarrow v$ are derivable from validated to be correct only is-a relations and/or mappings
   then remove (s,t) from $Repair(a,b)$;
5. return $Repair(a,b)$;

Fig. 4. Algorithm for generating repairing actions for missing is-a relations and mappings

In the network in Figure 1 *(nasal bone, bone)* in AMA is validated as missing. After adding the missing is-a relations to the ontology, its Source set is {*nasal bone, viscerocranium bone*} and its Target set is {*bone, limb bone, forelimb bone, hindlimb bone, foot bone, metatarsal bone, tarsal bone, jaw, maxilla, lacrimal bone*}, i.e., Repair contains $2 \times 10 = 20$ possible repairing actions.

Phase 3.2: Rank Wrong and Missing Is-a Relations and Mappings. In general, there will be many is-a relations/mappings that need to be repaired and some of them may be easier to start with such as the ones with fewer repairing actions. We therefore rank them with respect to the number of possible repairing actions.

Phase 3.3: Recommend Repairing Actions. The recommendation algorithm for wrong is-a relations/mappings assigns a priority to each possible repairing action based on how often it occurs in the justifications and its importance in already repaired is-a relations and mappings. For a missing is-a relation/mapping (a,b) (as defined in [19]) it computes the most informative repairing actions from $Source(a,b) \times Target(a,b)$ that are supported by external domain knowledge (WordNet and UMLS).

Phase 3.4: Execute Repairing Actions. Depending on whether a wrong or missing is-a relation/mapping is repaired the chosen repairing actions are removed from or added to the relevant ontologies and alignments. The current sets of wrong ($\mathcal{WI}$/$\mathcal{WM}$) and missing ($\mathcal{MI}$/$\mathcal{MM}$) is-a relations and mappings need to be updated since one repairing action can repair more than one is-a relation/mapping or previously repaired relations/mappings may need to be repaired again. The sets of repairing actions for wrong ($\mathcal{R}_I^-$, $\mathcal{R}_M^-$) and missing ($\mathcal{R}_I^+$, $\mathcal{R}_M^+$) is-a relations/mappings need to be updated as well. Further, new CMIs and CMMs may appear. In other cases the possible repairing actions for wrong and missing is-a relations and mappings may change (update justifications and sets of possible repairing actions for missing is-a relations and mappings). We also need to update the KBs.

5 Alignment Component

The input for this component consists of two taxonomies. The output is an alignment.

Phase 1: Detect Candidate Missing Mappings. In ontology alignment mapping suggestions are generated which essentially are CMMs. While the generation of CMMs in the debugging component is a specific kind of ontology alignment using the knowledge inherent in the network, in the alignment component we use other types of alignment algorithms. Matchers are used to compute similarity values between concepts in different ontologies. The results of the matchers can be combined and filtered in different ways to obtain mapping suggestions. In our approach we have currently used the linguistic, WordNet-based and UMLS-based algorithms from the SAMBO system [21]. The matcher *n-gram* computes a similarity based on 3-grams. The matcher *TermBasic* uses a combination of n-gram, edit distance and an algorithm that compares the lists of words of which the terms are composed. The matcher *TermWN* extends TermBasic by using WordNet for looking up is-a relations. The matcher *UMLSM* uses the domain knowledge in UMLS to obtain similarity values. The results of the matchers can be combined using a weighted-sum approach in which each matcher is given a weight and the final similarity value between a pair of concepts is the weighted sum of the similarity values divided by the sum of the weights of the used matchers. Further, we use a threshold for filtering. A pair of concepts is a mapping suggestion if the similarity value is equal to or higher than a given threshold value.

We note that in the alignment component the search space is not restricted to the *mapped concepts* only - similarity values are calculated for all pairs of concepts. KBs are initialized, in the same way as in the debugging component, for the taxonomy network and the pairs of taxonomies and their alignments. We also note that no initial alignment is needed for this component. Therefore, if alignments do not exist in the network (at all or between specific ontologies) this component may be used before starting debugging.

Phase 2: Validate Candidate Missing Mappings. The CMMs (mapping suggestions) are presented to a domain expert for validation, which is performed in the same way as in the debugging component. The domain expert can use the recommendation algorithms during the validation as well. As before, the CMMs are partitioned into two sets - wrong mappings and missing mappings. The wrong mappings are not repaired since they are not in the alignments. However, we store this information in order to avoid recomputations and for conflict checking/prevention. The concepts in the missing mappings are added to the set of *mapped concepts* (if they are not already there), and they will be used the next time CMMs/CMIs are derived in the debugging component.

Phase 3: Repairing Missing Mappings. As mentioned, we only need to repair the missing mappings. Initially, the missing mappings are added to the KBs in the same way as in the debugging component and then we try to repair them using more informative repairing actions. For repairing a missing mapping the same algorithms as in the debugging component are used to generate the Source and Target sets and the repairing process continues with the same actions described for the debugging workflow. In Phase 3.4 the repairing actions are executed analogically to those in the debugging

component and their consequences are computed. Further, the concepts in the repairing actions are added to the set of *mapped concepts* (if not there yet).

6 Interaction between the Components

The alignment component generates CMMs that are validated in the same way as in the debugging component. The CMMs validated to be correct often are missing mappings that are not found by the debugging component. Further, they may lead to new mapped concepts that are used in the debugging component. The CMMs validated to be wrong are used to avoid unnecessary recomputations and validations.

The debugging component repairs the is-a structure and the mappings. This can be used by the alignment component. For instance, the performance of structure-based matchers (e.g., [21]) and partial-alignment-based preprocessing and filtering methods [17] heavily depends on the correctness and completeness of the is-a structure.

We also note that the different phases in the components can be interleaved. This allows for an iterative and modular approach, where, for instance, some parts of the ontologies can be fully debugged and aligned before proceeding to other parts.

7 Experiments

We performed three experiments to demonstrate the benefits of the integrated ontology alignment and debugging framework. As input for Experiment 1 and 2 we used the two ontologies from the Anatomy track of OAEI 2011 - AMA contains 2,737 concepts and 1,807 asserted is-a relations, and NCI-A contains 3,298 concepts and 3,761 asserted is-a relations. The input for the last experiment contained the reference alignment (1516 equivalence mappings between AMA and NCI-A) together with the two ontologies. The reference alignment was used indirectly as external knowledge during the validation phase in the first two experiments. The experiments were performed on an Intel Core i7-2620M Processor 2.7GHz with 4 GB memory underWindows 7 Professional operating system and Java 1.7 compiler. The first author performed the validation in the experiments with help of two domain experts.

Experiment 1 - Aligning and Debugging OAEI Anatomy. The first experiment demonstrates a complete debugging and aligning session where the input is a set with the two ontologies. After loading the ontologies mapping suggestions were computed using matchers TermWN and UMLSM, weight 1 for both and threshold 0.5. This resulted in 1384 mapping suggestions. The 1233 mapping suggestions that are also in the reference alignment were validated as missing equivalence mappings (although, as we will see, there are defects in the reference alignment) and repaired by adding them to the alignment. The others were validated manually and resulted in missing mappings (53 equivalence and 39 is-a) and wrong mappings (59 equivalence and 39 is-a). These missing mappings were repaired by adding 53 equivalence and 28 is-a mappings (5 of them more informative) and 5 is-a relations (3 to AMA and 2 to NCI-A). 6 of these missing mappings were repaired by repairing others. Among the wrong mappings there were 3 which were derivable in the network. These were repaired by removing 2 is-a relations from NCI-A. Figure 5 - part A summarizes the results.

part A	candidate missing mappings	missing $\equiv$/$\leftarrow$ or $\rightarrow$	wrong $\equiv$/$\leftarrow$ or $\rightarrow$	repair missing $\equiv$/$\leftarrow$/$\rightarrow$/derivable /more informative	repair missing is-relations
Alignment	1384	1286/39	59/39	1286/20/8/6/5	-
AMA	-	-	-	-	3
NCI-A	-	-	-	-	2

part B	candidate missing all/non-redundant	missing	wrong	repair missing self/more informative/other	repair wrong removed
AMA	410/263	224	39	144/57/23	30
NCI-A	355/183	166	17	127/13/26	17
Alignment	-	-	-	-	8 $\equiv$ and 1 $\rightarrow$

Fig. 5. Experiment 1 results: A - debugging of the alignment; B - debugging of the ontologies

The generated alignment was then used in the debugging of the network created by the ontologies and the alignment. Two iterations of the debugging workflow were performed, since the repairing of wrong and missing is-a relations in the first iteration led to the detection of new CMIs which had to be validated and repaired. Over 90% of the CMIs for both ontologies were detected during the first iteration, the detection of CMIs took less than 30 seconds per ontology. Figure 5 - part B summarizes the results.

The system detected 410 (263 non-redundant) CMIs for AMA and 355 (183 non-redundant) CMIs for NCI-A. The non-redundant CMIs were displayed in groups, 45 groups for AMA and 31 for NCI-A. Among the 263 non-redundant CMIs in AMA 224 were validated as missing and 39 as wrong. In NCI-A 166 were validated as missing and 17 as wrong. The 39 wrong is-a relations in AMA were repaired by removing 30 is-a relations from NCI-A, and 8 equivalence and 1 is-a mapping from the alignment. The 17 wrong is-a relations in NCI-A were repaired by removing 17 is-a relations in AMA. The missing is-a relations in AMA were repaired by adding 201 is-a relations - in 144 cases the missing is-a relation itself and in 57 cases a more informative is-a relation. 23 of the 224 missing is-a relations became derivable after repairing some of the others. To repair the missing is-a relations in NCI-A 140 is-a relations were added - in 127 cases the missing is-a relation itself and in 13 cases a more informative is-a relation. 26 out of the 166 missing is-a relations were repaired while other is-a relations were repaired.

We observe that for 57 missing is-a relations in AMA and 13 in NCI-A the repairing actions are more informative than the missing is-a relation itself, i.e., for each of these, knowledge, which was not derivable from the network before, was added to the network. Thus the knowledge represented by the ontologies and the network has increased.

Experiment 2. For this experiment the alignment process was run twice and at the end the alignments were compared. The same matchers, weights and threshold as in Experiment 1 were used. During both runs the CMMs (mapping suggestions) were computed and validated in the same manner. This step is as in Experiment 1 and the results are the ones in Figure 5 - part A. The difference between both runs is in the repairing phase. In the first run the missing mappings were repaired by directly adding them to the final alignment without benefiting from the repairing algorithms - in the same way most of

the alignment systems do. The final alignment contained 1286 equivalence and 39 is-a[5] mappings.

During the repairing phase in the second run the debugging component was used to provide alternative repairing actions than those available in the initial set of mapping suggestions. The final alignment then contained 1286 equivalence mappings from the mapping suggestions, 28 is-a mappings from the mapping suggestions where 5 of them are more informative, thus adding knowledge to the network. Further, 5 mapping suggestions were repaired adding is-a relations (3 in AMA and 2 in NCI-A) and thus adding more knowledge to each of the ontologies. 6 more mapping suggestions became derivable from the network as a result from the repairing actions for other CMMs.

Experiment 3. In this experiment the debugging process was run twice, CMIs were detected for both ontologies and compared between the runs. The input for the first run was the set of the two ontologies and their alignment from the Anatomy track in OAEI 2011. The network was loaded in the system and the CMIs were detected. 496 CMIs were detected for AMA, of which 280 were non-redundant. For NCI-A 365 CMIs were detected of which 193 were non-redundant. The same input was used in the second run. However, the alignment algorithms were used to extend the set with mappings prior to generating the CMIs. The set-up for the aligning was the same as in Experiment 1 and the mapping suggestions were computed, validated and repaired in the same way as well. Then CMIs were generated - 638 CMIs were detected for AMA (357 non-redundant), and 460 CMIs for NCI-A (234 non-redundant). In total 145 new CMIs were detected for AMA - 120 were validated as missing and 25 validated as wrong[6]. 103 new CMIs were detected for NCI-A - 53 were validated as missing and 50 as wrong.

Discussion. Experiment 1 shows the usefulness of the system through a complete session where an alignment was generated and many defects in the ontologies were repaired. Some of the repairs added new knowledge. As a side effect, we have shown that the ontologies that are used by the OAEI contain over 200 and 150 missing is-a relations, respectively and 39 and 17 wrong is-a relations, respectively. We have also shown that the alignment is not complete and contains wrong information. We also note that our system allows validation and allows a domain expert to distinguish between equivalence and is-a mappings. Most ontology alignment systems do not support this.

Experiment 2 shows the advantages for ontology alignment when also a debugging component is added. The debugging component allowed to add more informative mappings, reduce redundancy in the alignment as well as debug the ontologies leading to further reduced redundancy in the alignment. For the ontologies and alignment new knowledge not found when only aligning, was added. In general, the quality of the final alignment (and the ontologies) becomes higher.

Experiment 3 shows that the debugging process can take advantage of the alignment component even when an alignment is available. The alignment algorithms can provide additional mapping suggestions and thus extending the alignment. More mappings be-

[5] 5 of these are repaired in the second run by adding is-a relations in the ontologies.

[6] The sum of the newly generated CMIs and those in the first run is not equal to the number of the CMIs in the second run because some of the CMIs generated in the first run are derivable in the second run.

tween two ontologies means higher coverage and possibly more detected and repaired defects. In the experiment more than 100 CMIs (of which many correct) were detected for each ontology using the extended set of mappings. We also note that the initial alignment contained many mappings (1516). In the case that the alignment contains fewer mappings the benefit to the debugging process will be even more significant.

8 Related Work

To our knowledge there is no other system that integrates ontology debugging and ontology alignment in a uniform way and that allows for a strong interleaving of these tasks. There are some ontology alignment systems that do semantic verification and disallow mappings that lead to unsatisfiable concepts (e.g., [10,12]). Further, adding missing is-a relations to ontologies was a step in the alignment process in [17].

Regarding the debugging component, this work extends the work in [19,18] that dealt with debugging is-a structure in taxonomy networks. These were one of the few approaches dealing with repairing missing is-a structure and in the case of [18] debugging both missing and wrong is-a structure. The current work extends this by also including debugging of mappings in a uniform way as well as ontology alignment. The ontology alignment component also removed the restriction of [18] that required the existence of an initial alignment.

There are different ways to *detect* missing is-a relations. One way is by inspection of the ontologies by domain experts. Another way is to use external knowledge sources. For instance, there is much work on finding relationships between terms in the ontology learning area [1]. Regarding the detection of is-a relations, one paradigm is based on linguistics using lexico-syntactic patterns. The pioneering research conducted in this line is in [9], which defines a set of patterns indicating is-a relationships between words in the text. Another paradigm is based on machine learning and statistical methods. Further, guidelines based on logical patterns can be used [3]. These approaches are complementary to the approach used in this paper. There is, however, not much work on the *repairing* of missing is-a relations that goes beyond adding them to the ontologies except for [19] for taxonomies and [16] for $\mathcal{ALC}$ acyclic terminologies.

There is more work on the debugging of semantic defects. Most of it aims at identifying and removing logical contradictions from an ontology. Standard reasoners are used to identify the existence of a contradiction, and provide support for resolving and eliminating it [6]. In [24] minimal sets of axioms are identified which need to be removed to render an ontology coherent. In [15,14] strategies are described for repairing unsatisfiable concepts detected by reasoners, explanation of errors, ranking erroneous axioms, and generating repair plans. In [8] the focus is on maintaining the consistency as the ontology evolves through a formalization of the semantics of change for ontologies. [26] introduces a method for interactive ontology debugging. In [22] and [11] the setting is extended to repairing ontologies connected by mappings. In this case, semantic defects may be introduced by integrating ontologies. Both works assume that ontologies are more reliable than the mappings and try to remove some of the mappings to restore consistency. The solutions are often based on the computation of minimal unsatisfiability-preserving sets or minimal conflict sets. The work in [23] further characterizes the problem as mapping revision. Using belief revision theory, the authors

give an analysis for the logical properties of the revision algorithms. Another approach for debugging mappings is proposed in [28] where the authors focus on the detection of certain kinds of defects and redundancy. The approach in [13] deals with the inconsistencies introduced by the integration of ontologies, and unintended entailments validated by the user.

Regarding the alignment component there are some systems that allow validation of mappings such as SAMBO [21], COGZ [5] for PROMPT, and COMA++ [4]. [7] introduces an efficient algorithm for computing a minimal set with mappings which could reduce user interaction. Many matchers have been proposed (e.g., many papers on http://ontologymatching.org/), and most systems use similar combination and filtering strategies as in this paper. For an overview we refer to [25].

9 Conclusion

In this paper we presented a unified approach for aligning taxonomies and debugging taxonomies and their alignments. This is the first approach which integrates ontology alignment and ontology debugging and allows debugging of both the structure of the ontologies as well as their alignments. Further, we have shown the benefits of our approach through experiments. The interactions between ontology alignment and debugging significantly raise the quality of both taxonomies and their alignments. The ontology alignment provides or extends alignments that are used by the debugging. The debugging provides algorithms for repairing defects in alignments and possibly add new knowledge.

We will continue exploring the interactions between ontology alignment and debugging. We will include and investigate the benefits when using structure-based alignment algorithms and partial-alignment-based techniques. Further, we will investigate the debugging problem for ontologies represented in more expressive formalisms.

Acknowledgements. We thank the Swedish Research Council (Vetenskapsrådet) and the Swedish e-Science Research Centre (SeRC) for financial support.

References

1. Cimiano, P., Buitelaar, P., Magnini, B.: Ontology Learning from Text: Methods, Evaluation and Applications. IOS Press (2005)
2. Conroy, C., Brennan, R., O'Sullivan, D., Lewis, D.: User Evaluation Study of a Tagging Approach to Semantic Mapping. In: Aroyo, L., et al. (eds.) ESWC 2009. LNCS, vol. 5554, pp. 623–637. Springer, Heidelberg (2009)
3. Corcho, O., Roussey, C., Vilches, L.M., Pérez, I.: Pattern-based OWL ontology debugging guidelines. In: Workshop on Ontology Patterns, pp. 68–82 (2009)
4. Do, H.-H., Rahm, E.: Matching large schemas: approaches and evaluation. Information Systems 32, 857–885 (2007)
5. Falconer, S.M., Storey, M.-A.: A cognitive support framework for ontology mapping. In: Aberer, K., et al. (eds.) ISWC/ASWC 2007. LNCS, vol. 4825, pp. 114–127. Springer, Heidelberg (2007)
6. Flouris, G., Manakanatas, D., Kondylakis, H., Plexousakis, D., Antoniou, G.: Ontology change: Classification and survey. Knowledge Engineering Review 23(2), 117–152 (2008)

7. Giunchiglia, F., Maltese, V., Autayeu, A.: Computing minimal mappings. In: Ontology Matching Workshop, pp. 37–48 (2009)
8. Haase, P., Stojanovic, L.: Consistent Evolution of OWL Ontologies. In: Gómez-Pérez, A., Euzenat, J. (eds.) ESWC 2005. LNCS, vol. 3532, pp. 182–197. Springer, Heidelberg (2005)
9. Hearst, M.: Automatic acquisition of hyponyms from large text corpora. In: 14th International Conference on Computational Linguistics, pp. 539–545 (1992)
10. Jean-Mary, Y.R., Shironoshita, E.P., Kabuka, M.R.: Ontology matching with semantic verification. Journal of Web Semantics 7(3), 235–251 (2009)
11. Ji, Q., Haase, P., Qi, G., Hitzler, P., Stadtmüller, S.: RaDON — repair and diagnosis in ontology networks. In: Aroyo, L., et al. (eds.) ESWC 2009. LNCS, vol. 5554, pp. 863–867. Springer, Heidelberg (2009)
12. Jimenez-Ruiz, E., Cuenca-Grau, B., Zhou, Y., Horrocks, I.: Large-scale interactive ontology matching: Algorithms and implementation. In: 20th European Conference on Artificial Intelligence, pp. 444–449 (2012)
13. Jiménez-Ruiz, E., Cuenca Grau, B., Horrocks, I., Berlanga, R.: Ontology integration using mappings: Towards getting the right logical consequences. In: Aroyo, L., et al. (eds.) ESWC 2009. LNCS, vol. 5554, pp. 173–187. Springer, Heidelberg (2009)
14. Kalyanpur, A., Parsia, B., Sirin, E., Cuenca-Grau, B.: Repairing Unsatisfiable Concepts in OWL Ontologies. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, pp. 170–184. Springer, Heidelberg (2006)
15. Kalyanpur, A., Parsia, B., Sirin, E., Hendler, J.: Debugging Unsatisfiable Classes in OWL Ontologies. Journal of Web Semantics 3(4), 268–293 (2006)
16. Lambrix, P., Dragisic, Z., Ivanova, V.: Get my pizza right: Repairing missing is-a relations in ALC ontologies. In: 2nd Joint International Semantic Technology Conference (2012)
17. Lambrix, P., Liu, Q.: Using partial reference alignments to align ontologies. In: Aroyo, L., et al. (eds.) ESWC 2009. LNCS, vol. 5554, pp. 188–202. Springer, Heidelberg (2009)
18. Lambrix, P., Liu, Q.: Debugging is-a structure in networked taxonomies. In: 4th International Workshop on Semantic Web Applications and Tools for Life Sciences, pp. 58–65 (2011)
19. Lambrix, P., Liu, Q., Tan, H.: Repairing the missing is-a structure of ontologies. In: Gómez-Pérez, A., Yu, Y., Ding, Y. (eds.) ASWC 2009. LNCS, vol. 5926, pp. 76–90. Springer, Heidelberg (2009)
20. Lambrix, P., Qi, G., Horridge, M.: Proceedings of the 1st International Workshop on Debugging Ontologies and Ontology Mappings. LiU E-Press, LECP 79 (2012)
21. Lambrix, P., Tan, H.: SAMBO - a system for aligning and merging biomedical ontologies. Journal of Web Semantics 4(3), 196–206 (2006)
22. Meilicke, C., Stuckenschmidt, H., Tamilin, A.: Repairing Ontology Mappings. In: 22nd Conference on Artificial Intelligence, pp. 1408–1413 (2007)
23. Qi, G., Ji, Q., Haase, P.: A Conflict-Based Operator for Mapping Revision. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 521–536. Springer, Heidelberg (2009)
24. Schlobach, S.: Debugging and Semantic Clarification by Pinpointing. In: Gómez-Pérez, A., Euzenat, J. (eds.) ESWC 2005. LNCS, vol. 3532, pp. 226–240. Springer, Heidelberg (2005)
25. Schvaiko, P., Euzenat, J.: Ontology matching: state of the art and future challenges. IEEE Transactions on Knowledge and Data Engineering 25(1), 158–176 (2013)
26. Shchekotykhin, K., Friedrich, G., Fleiss, P., Rodler, P.: Interactive ontology debugging: Two query strategies for efficient fault localization. Journal of Web Semantics 12-13, 88–103 (2012)
27. UMLS. Unified medical language system, `http://www.nlm.nih.gov/research/umls/about_umls.html`
28. Wang, P., Xu, B.: Debugging ontology mappings: a static approach. Computing and Informatics 27, 21–36 (2008)
29. WordNet, `http://wordnet.princeton.edu/`