Interactive Ontology Debugging using Direct Diagnosis

Kostyantyn Shchekotykhin, Gerhard Friedrich, Patrick Rodler, and Philipp Fleiss

Alpen-Adria Universität, Klagenfurt, 9020 Austria, email: firstname.lastname@aau.at

Abstract. Sequential diagnosis methods compute a series of queries for discriminating between diagnoses. Queries are answered by some oracle such that eventually the set of faults is identified. The computation of queries is based on the generation of a set of *most probable* diagnoses. However, in diagnosis problem instances where the number of minimal diagnoses and their cardinality is high, even the generation of a set of minimum cardinality diagnoses is unfeasible with the standard conflict-based approach. In this paper we propose to base sequential diagnosis on the computation of *some* set of minimal diagnoses using the direct diagnosis method, which requires less consistency checks to find a minimal diagnosis than the standard approach. We study the application of this direct method to high cardinality faults in ontologies. In particular, our evaluation shows that the direct method results in almost the same number of queries for cases when the standard approach is applicable. However, for the cases when the standard approach is not applicable, sequential diagnosis based on the direct method is able to locate the faults correctly.

1 Introduction

Standard sequential model-based diagnosis (MBD) methods [18, 15] acquire additional information in order to discriminate between diagnoses. Queries are generated and answered either by automatic probing or by asking humans for additional observations about the system to be diagnosed. As various applications show, the standard methods work very satisfactorily for cases where the number of faults is low (single digit number), consistency checking is fast (single digit number of seconds), and sufficient possibilities for observations are available for discriminating between diagnoses.

MBD is a general method which can be used to find errors in hardware, software, knowledge-bases, orchestrated web-services, configurations, etc. In particular, OWL ontology debugging tools [14, 7, 10] can localize a (potential) fault by computing sets of axioms $\mathcal{D} \subseteq \mathcal{O}$ called *diagnosis* for an ontology $\mathcal{O}$. At least all axioms of a diagnosis must be modified or deleted in order to formulate a fault-free ontology $\mathcal{O}^*$. The latter is faulty if some requirements, such as consistency of $\mathcal{O}$, presence or absence of specific entailments, are violated.

All the discrimination and diagnosis approaches listed above follow the standard MBD technique [18] and compute diagnoses using minimal conflict sets, i.e. irreducible sets of axioms $CS \subseteq \mathcal{O}$ that violate some requirements, by using a consistency checker (black-box approach). Furthermore, diagnoses are ordered and filtered by some preference criteria, e.g. probability or cardinality, in order to focus debugging on the most likely cases.

In the common ontology development scenario where a user develops an ontology manually, the changes between validation steps, e.g. consistency checking, are rather small. Therefore, the number of faulty axioms is in a range where standard sequential MBD methods are applicable [20]. However, there are cases when the changes are substantial. For example, in ontology matching two ontologies with several thousands of axioms are merged into a single one. High quality matchers (e.g. [12]) require the diagnosis of such merged ontologies, but often cannot apply standard MBD methods because of the large number of minimum cardinality diagnoses and their high cardinality (e.g. greater than 20). This observation is supported by analysis of justifications [11], which is a dual problem to computation of diagnoses. Moreover, most of the diagnostic problems are NP-complete even if reasoning is done in polytime [4, 17].

In order to deal with hard diagnosis instances, we propose to relax the requirement for sequential diagnosis to compute a set of *preferred* diagnoses, such as a set of most probable diagnoses. Instead, we compute *some* set of diagnoses which can be employed for query generation. This allows to use the direct computation of diagnoses [19] *without* computing conflict sets. The direct approach was applied for non-interactive diagnosis of ontologies [3, 2] and constraints [6]. The computation of a diagnosis $\mathcal{D}$ by a variant of QUICKXPLAIN [13] requires $O(|\mathcal{D}|\log(\frac{|\mathcal{O}|}{|\mathcal{D}|}))$ consistency checks, where $|\mathcal{D}|$ is the cardinality of the diagnosis and $|\mathcal{O}|$ the size of the knowledge base. If m diagnoses are required for query generation, then only m calls to a direct diagnosis generator are needed. A recent approach [21] does not generate the standard HS-TREE, but still depends on the minimization of conflict sets, i.e. $|\mathcal{D}|$ minimized conflicts have to be discovered. Consequently, if $|\mathcal{D}| \gg m$, substantially more consistency checks are required.

Since we are replacing the set of most probable diagnoses by just a set of diagnoses, some important practical questions have to be addressed. (1) Is a substantial number of additional queries needed, (2) is this approach able to locate the faults, and (3) how efficient is this approach?

In order to answer these questions we have exploited the most difficult diagnoses problems of the ontology alignment competition [5]. Our evaluation shows that sequential diagnosis by direct diagnosis generation needs approximately the same number of queries (± 1) in order to identify the faults. This evaluation was carried out for cases where the standard sequential diagnosis method was applicable. Furthermore, the evaluation shows that our proposed direct method is capable of locating faults in all cases correctly. Moreover, for the hardest instance the computation costs which are introduced in addition to the computational costs of theorem proving are less than 7%.

The remainder of the paper is organized as follows: Section 2 gives a brief introduction to the main notions of sequential ontology diagnosis. The details of the suggested algorithms and their applications are presented in Section 3. In Section 4 we provide evaluation results.

2 Basic concepts

In the following we present (1) the fundamental concepts regarding the diagnosis of ontologies and (2) the interactive localization of axioms that must be changed.

Diagnosis of ontologies. Given an ontology $\mathcal{O}$ which is a set of logical sentences (axioms), the user can specify particular requirements during the knowledge-engineering process. The most basic requirement is consistency, i.e. a logical model exists. A further frequently employed requirement is coherence. In addition, as it is common practice in software engineering, the knowledge-engineer (user for short) may specify test cases, which are axioms which must (not) be entailed by a valid ontology.

Given a set of axioms P (positive test cases) and a set of axioms N (negative test cases), an ontology $\mathcal{O}^*$ is valid iff $\mathcal{O}^*$ is consistent (and coherent if required) and

1. $\mathcal{O}^* \models p$ *for all* $p \in P$
2. $\mathcal{O}^* \not\models n$ *for all* $n \in N$

Let us assume that there is a non-valid ontology $\mathcal{O}$, then a set of axioms $\mathcal{D} \subseteq \mathcal{O}$ must be removed and possibly some axioms EX must be added by a user s.t. an updated $\mathcal{O}^*$ becomes valid, i.e. $\mathcal{O}^* := (\mathcal{O} \setminus \mathcal{D}) \cup EX$. The goal of diagnosis is to provide information which sets of axioms $\mathcal{D}$ should be revised in order to formulate a valid ontology. Furthermore, we allow the user to define a set of axioms $\mathcal{B}$ (called the background theory) which must not be changed (i.e. the correct axioms).

Definition 1. *Let $\langle \mathcal{O}, \mathcal{B}, P, N \rangle$ be a diagnosis problem instance (DPI) where $\mathcal{O}$ is a ontology, $\mathcal{B}$ a background theory, P a set of axioms which must be implied by a valid ontology $\mathcal{O}^*$, and N a set of axioms which must* not *be implied by $\mathcal{O}^*$.*

A set of axioms $\mathcal{D} \subseteq \mathcal{O}$ is a candidate diagnosis iff the set of axioms $\mathcal{O} \setminus \mathcal{D}$ can be extended by a set of logical sentences EX such that:

1. *$(\mathcal{O} \setminus \mathcal{D}) \cup \mathcal{B} \cup EX$ is consistent (and coherent if required)*
2. *$(\mathcal{O} \setminus \mathcal{D}) \cup \mathcal{B} \cup EX \models p$ for all $p \in P$*
3. *$(\mathcal{O} \setminus \mathcal{D}) \cup \mathcal{B} \cup EX \not\models n$ for all $n \in N$*

$\mathcal{D}$ is a diagnosis *iff there is no $\mathcal{D}' \subset \mathcal{D}$ such that $\mathcal{D}'$ is a candidate diagnosis. $\mathcal{D}$ is a* minimum cardinality diagnosis *iff there is no diagnosis $\mathcal{D}'$ such that $|\mathcal{D}'| < |\mathcal{D}|$.*

The following proposition of [20] characterizes diagnoses by replacing EX with the positive test cases.

Corollary 1. *Given a DPI $\langle \mathcal{O}, \mathcal{B}, P, N \rangle$, a set of axioms $\mathcal{D} \subseteq \mathcal{O}$ is a diagnosis iff $(\mathcal{O} \setminus \mathcal{D}) \cup \mathcal{B} \cup \{\bigwedge_{p \in P} p\}$ is consistent (coherent) and $\forall n \in N : (\mathcal{O} \setminus \mathcal{D}) \cup \mathcal{B} \cup \{\bigwedge_{p \in P} p\} \not\models n$*

In the following we assume that there is always a diagnosis.

Proposition 1. *A diagnosis $\mathcal{D}$ for a DPI $\langle \mathcal{O}, \mathcal{B}, P, N \rangle$ exists iff $\mathcal{B} \cup \{\bigwedge_{p \in P} p\}$ is consistent (coherent) and $\forall n \in N : \mathcal{B} \cup \{\bigwedge_{p \in P} p\} \not\models n$*

For the computation of diagnoses *conflict sets* are usually employed to constrain the search space. A conflict set is the part of the ontology that preserves the inconsistency/incoherency.

Definition 2. *Given a DPI $\langle \mathcal{O}, \mathcal{B}, P, N \rangle$, a set of axioms $CS \subseteq \mathcal{O}$ is a conflict set iff $CS \cup \mathcal{B} \cup \{\bigwedge_{p \in P} p\}$ is inconsistent (incoherent) or there is an $n \in N$ s.t. $CS \cup \mathcal{B} \cup \{\bigwedge_{p \in P} p\} \models n$. CS is a minimal conflict set iff there is no $CS' \subset CS$ such that CS' is a conflict set.*

Minimal conflict sets can be used to compute the set of diagnoses as it is shown in [18]. The idea is that each diagnosis should include at least one element of each minimal conflict set.

Proposition 2. *$\mathcal{D}$ is a diagnosis for the DPI $\langle \mathcal{O}, \mathcal{B}, P, N\rangle$ iff $\mathcal{D}$ is a minimal hitting set for the set of all minimal conflict sets of the DPI.*

Generation of a minimal conflict set is done by specific algorithms such as the divide-and-conquer method QUICKXPLAIN (QX) [13]. In the worst case, QX requires $O(|CS| \log(\frac{|\mathcal{O}|}{|CS|}))$ calls to the reasoner, where CS is the returned minimal conflict set.

The computation of diagnoses in ontology debugging systems is implemented using Reiter's Hitting Set HS-TREE algorithm [18]. The algorithm constructs a directed tree from the root to the leaves, where each non-leave node is labeled with a minimal conflict set and leave nodes are labeled by ✓ (*no conflicts*) or $\times$ (*pruned*).

Each (✓) node corresponds to a diagnosis. The subset minimality of the diagnoses is guaranteed by the minimality of conflict sets used for labeling the nodes, the pruning rule and the breadth-first strategy for tree generation [18]. Moreover, because of the breadth-first strategy the diagnoses are generated in increasing order of their cardinality. Under the assumption that diagnoses with lower cardinality are more probable than those with higher cardinality, HS-TREE generates most probable diagnoses first.

Diagnoses discrimination. For many real-world DPIs, an ontology debugger can return a large number of diagnoses. Each diagnosis corresponds to a different set of axioms that must be changed in order to formulate a valid ontology. The user may extend the test cases P and N s.t. diagnoses are eliminated, thus identifying exactly those axioms that must be changed. That is, we assume that the user (oracle) is equipped with sufficient knowledge about the valid ontology $\mathcal{O}^*$ such that axiom Q can be classified either as entailed by $\mathcal{O}^*$ or not. If a user finds that Q must be entailed by $\mathcal{O}^*$, then it is added to the set P yielding the new DPI $\langle \mathcal{O}, \mathcal{B}, P \cup \{Q\}, N\rangle$, and to N, i.e. $\langle \mathcal{O}, \mathcal{B}, P, N \cup \{Q\}\rangle$, otherwise. According to Definition 1, any diagnosis of the original DPI is not a diagnosis of an updated DPI if it violates any of its test cases. Moreover, in case $Q \in \mathcal{O}$, each diagnosis of an updated DPI must comprise Q if $Q \in N$ and not comprise Q if $Q \in P$.

Property 1. Given set of diagnoses $\mathbf{D}$ for a DPI $\langle \mathcal{O}, \mathcal{B}, P, N\rangle$ and an axiom Q representing the oracle query $\mathcal{O}^* \models Q$. If the oracle gives the answer *yes* then every diagnosis $\mathcal{D}_i \in \mathbf{D}$ is a diagnosis for $\langle \mathcal{O}, \mathcal{B}, P \cup \{Q\}, N\rangle$ iff both conditions hold:

$$(\mathcal{O} \setminus \mathcal{D}_i) \cup \mathcal{B} \cup \{\bigwedge_{p \in P} p\} \cup \{Q\} \textit{ is consistent (coherent)}$$

$$\forall n \in N \ : \ (\mathcal{O} \setminus \mathcal{D}_i) \cup \mathcal{B} \cup \{\bigwedge_{p \in P} p\} \cup \{Q\} \not\models n$$

If the oracle gives the answer *no* then every diagnosis $\mathcal{D}_i \in \mathbf{D}$ is a diagnosis for $\langle \mathcal{O}, \mathcal{B}, P, N \cup \{Q\}\rangle$ iff both conditions hold:

$$(\mathcal{O} \setminus \mathcal{D}_i) \cup \mathcal{B} \cup \{\bigwedge_{p \in P} p\} \textit{ is consistent (coherent)}$$

$$\forall n \in (N \cup \{Q\}) \ : \ (\mathcal{O} \setminus \mathcal{D}_i) \cup \mathcal{B} \cup \{\bigwedge_{p \in P} p\} \not\models n$$

However, many different queries might exist for some set of diagnoses $|\mathbf{D}| > 2$, in the extreme case exponentially many (in $|\mathbf{D}|$). To select the best query, the authors in [20] suggest two query selection strategies: SPLIT-IN-HALF (SPL) and ENTROPY (ENT). The first strategy is a greedy approach preferring queries which allow to remove half of the diagnoses in $\mathbf{D}$, for both answers to the query. The second is an information-theoretic measure, which estimates the information gain for both outcomes of each query and returns the query that maximizes the expected information gain. The *prior fault probabilities* required for evaluating the ENTROPY measure can be obtained from statistics of previous diagnosis sessions. For instance, if the user has problems to apply "$\exists$", then the diagnosis logs are likely to contain more repairs of axioms including this quantifier. Consequently, the prior fault probabilities of axioms including "$\exists$" should be higher. Given the fault probabilities of axioms, one can calculate prior fault probabilities of diagnoses as well as evaluate ENTROPY (see [20] for more details). The queries for both strategies are constructed by exploiting so called classification and realization services provided by description logic reasoners. Given a ontology $\mathcal{O}$ the classification generates the subsumption hierarchy, i.e. the entailments $\mathcal{O} \models A \sqsubseteq B$, where B is the most specific concept that subsumes A. Realization computes, for each individual name t occurring in an ontology $\mathcal{O}$, a set of most specific concepts A s.t. $\mathcal{O} \models A(t)$ (see [1] for details).

Due to the number of diagnoses and the complexity of diagnosis computation, not all diagnoses are exploited for generating queries but a set of (most probable) diagnoses of size less or equal to some (small) predefined number m [20]. We call this set the *leading diagnoses* and denote it by $\mathbf{D}$ from now on. The set of leading diagnoses $\mathbf{D}$ acts as a representative of the set of all diagnoses.

The *standard* sequential ontology debugging process can be sketched as follows. As input a DPI and some meta information, i.e. prior fault estimates $\mathcal{F}$, a query selection strategy s_Q (SPL or ENT) and a stop criterion σ, are given. As output a diagnosis is returned that has a posterior probability of at least $1 - \sigma$. For sufficiently small σ this means that the returned diagnosis is highly probable whereas all other leading diagnoses are highly improbable.

1. Using QX and HS-TREE (re-)calculate a set of leading diagnoses $\mathbf{D}$ of cardinality $\min(m, a)$, where a is the number of all diagnoses for the DPI and m is the number of leading diagnoses predefined by a user.
2. Use the prior fault probabilities $\mathcal{F}$ and the already specified test cases to compute (posterior) probabilities of diagnoses in $\mathbf{D}$ by the Bayesian Rule (cf. [20]).
3. If some diagnosis $\mathcal{D} \in \mathbf{D}$ has probability greater or equal to $1 - \sigma$ or the user accepts $\mathcal{D}$ as the axioms to be changed then stop and return $\mathcal{D}$.
4. Use $\mathbf{D}$ to generate a set of queries and select the best query Q according to s_Q.
5. Ask the user $\mathcal{O}^* \models Q$ and, depending on the answer, add Q either to P or to N.
6. Remove elements from $\mathbf{D}$ violating the newly acquired test case.
7. Repeat at Step 1.

3 Interactive Direct Diagnosis of Ontologies

The novelty of our approach is the interactivity combined with the direct calculation of diagnoses. To this end, we provide modifications to Step 1 of the diagnosis process given above. Namely, we utilize an "inverse" version of the QX algorithm [13] called INV-QX and an associated "inverse" version of HS-TREE termed INV-HS-TREE.

This combination of algorithms was first used in the earlier version of [6]. However, we introduced two modifications: (i) a depth-first search strategy instead of breadth-first and (ii) a new pruning rule which moves axioms from $\mathcal{O}$ to $\mathcal{B}$ instead of just removing them from $\mathcal{O}$, since not adding them to $\mathcal{B}$ might result in losing some of the diagnoses.

INV-QX – Key Idea. INV-QX relies on the monotonic semantics of the used knowledge representation language. The algorithm takes a DPI $\langle \mathcal{O}, \mathcal{B}, P, N \rangle$ and a ranking heuristic as input and outputs either one diagnosis or *no-diagnosis-exists*. The ranking heuristic assigns a fault probability to each axiom in $\mathcal{O}$, if this information is available; otherwise every axiom has the same rank. In the first step INV-QX verifies if a diagnosis exists, next whether $\mathcal{O}$ is faulty and, if so, sorts all axioms in descending order. Ordering of axioms according to their fault probabilities allows the algorithm to compute an approximation of a most probable diagnosis. Next, INV-QX enters the recursion in which $\mathcal{O}$ is partitioned into two subsets S_1 and S_2 such that S_1 comprises axioms with higher fault probabilities and S_2 with lower. In our implementation $\mathcal{O}$ is split in half. Then the algorithm verifies whether S_1 is a candidate diagnosis of the input DPI according to Definition 1. The algorithm continues to operate in a divide-and-conquer strategy until a diagnosis is found. INV-QX requires $O(|\mathcal{D}| \log(\frac{|\mathcal{O}|}{|\mathcal{D}|}))$ calls to a reasoner to find a diagnosis $\mathcal{D}$.

INV-QX is a deterministic algorithm. In order to obtain a different next diagnosis, the DPI used as input for INV-QX must be modified accordingly. To this end we employ INV-HS-TREE.

INV-HS-TREE – Construction. The algorithm is inverse to the HS-TREE algorithm in the sense that nodes are now labeled by diagnoses (instead of minimal conflict sets) and a path from the root to an open node is a partial conflict set (instead of a partial diagnosis). The algorithm constructs a directed tree from the root to the leaves, where each node nd is labeled either with a diagnosis $\mathcal{D}$ or $\times$ (*pruned*) which indicates that the node is closed. For each $s \in \mathcal{D}$ there is an outgoing edge labeled by s. Let $H(nd)$ be the set of edge labels on the path from the root to the node nd. Initially the algorithm generates an empty root node and adds it to a LIFO-queue, thereby implementing a *depth-first search* strategy. Until the required number m of diagnoses is reached or the queue is empty, the algorithm removes the first node nd from the queue and labels the node by applying the following steps.

1. (*reuse*): if $\mathcal{D} \cap H(nd) = \emptyset$ for some $\mathcal{D} \in \mathbf{D}$, then label the node with $\mathcal{D}$; add for each $s \in \mathcal{D}$ a node to the LIFO-queue and return
2. Call INV-QX$(\mathcal{O} \setminus H(nd), \mathcal{B} \cup H(nd), P, N) = Value$
3. (*prune*): if $Value =$ *no-diagnosis-exists*, then label the node with $\times$ (see Proposition 1) and return
4. (*assign*): Otherwise $Value$ is a diagnosis, label the node with $\mathcal{D} = Value$; add $\mathcal{D}$ to $\mathbf{D}$ and add for each $s \in \mathcal{D}$ a node to the LIFO-queue.

Reuse of known diagnoses in Step 1 and the addition of $H(nd)$ to the background theory $\mathcal{B}$ in Step 2 allows the algorithm to force INV-QX to search for a diagnosis that is different to all diagnoses in $\mathbf{D}$. In case INV-QX returns *no-diagnosis-exists* the node is pruned. Otherwise, a new diagnosis $\mathcal{D}$ is added to the set $\mathbf{D}$ and is used to label the node. The depth-first search strategy maintains only a set of diagnoses comprising at most m elements. No conflicts are stored. This allows a significant reduction of memory usage by INV-HS-TREE compared to HS-TREE. The worst case space complexity of INV-HS-TREE computing m diagnoses is *linear* and amounts to $O(m)$, whereas the worst case space complexity of HS-TREE is $O(|CS_{\max}|^d)$ where $|CS_{\max}|$ is the maximal cardinality minimal conflict set (i.e. there is no minimal conflict set with larger cardinality) and d is the depth were m diagnoses have been generated w.r.t. a DPI.

The disadvantage of INV-HS-TREE is that it cannot guarantee the computation of diagnoses in a special order, e.g. minimum cardinality or maximum probability first.

INV-HS-TREE – Update Procedure for Interactivity. Since paths in INV-HS-TREE are (1) irrelevant and need not be maintained, and (2) only a small (linear) number of nodes/paths is in memory due to the application of a depth-first search, the update procedure after a query Q has been answered involves a reconstruction of the tree. In particular, by answering Q, $m-k$ of (maximally) m leading diagnoses are invalidated and deleted from memory. The k still valid diagnoses are used to build a new tree. To this end, the root is labeled by any of these k diagnoses and a tree is constructed as described above where the k diagnoses are incorporated for the *reuse* check. Note, the recalculation of a diagnosis that has been invalidated by a query is impossible as in subsequent iterations a new DPI is considered which includes the answered query as a test case.

Example. Consider a DPI with $\mathcal{O} = \{ax_1 : C \sqsubseteq A \quad ax_2 : C \sqsubseteq E \quad ax_3 : A \sqsubseteq \neg(C \sqcup \neg B) \quad ax_4 : B \sqsubseteq C \quad ax_5 : B \sqsubseteq \neg D\}$ the background knowledge $\mathcal{B} = \{A(v), B(w), C(s)\}$, one positive $P = \{D(v)\}$ and one negative $N = \{E(w)\}$ test case. For the sample DPI the set of minimal conflict sets comprises four elements $\{CS_1 : \langle ax_1, ax_3\rangle, CS_2 : \langle ax_2, ax_4\rangle, CS_3 : \langle ax_3, ax_5\rangle, CS_4 : \langle ax_3, ax_4\rangle\}$, as well as the set of diagnoses $\{\mathcal{D}_1 : [ax_2, ax_3], \mathcal{D}_2 : [ax_3, ax_4], \mathcal{D}_3 : [ax_1, ax_4, ax_5]\}$. Assume also that the number of leading diagnoses required for query generation is set to $m = 2$. Applied to the sample DPI, INV-HS-TREE computes a diagnosis $\mathcal{D}_1 := [ax_2, ax_3]$ returned by INV-QX$(\mathcal{O}, \mathcal{B}, P, N)$ to label the root node, see Figure 1. Next, it generates one successor node that is linked with the root by an edge labeled with ax_2. For this node INV-QX$(\mathcal{O} \setminus \{ax_2\}, \mathcal{B} \cup \{ax_2\}, P, N)$ yields a diagnosis $\mathcal{D}_2 := [ax_3, ax_4]$ disjoint with $\{ax_2\}$. Now $|\mathbf{D}| = 2$ and a query is generated and answered as in Figure 1. Adding $C(w)$ to the negative test cases invalidates $\mathcal{D}_1$ because of $ax_4 \in (\mathcal{O} \setminus \mathcal{D}_1)$ and $B(w) \in \mathcal{B}$, that is $(\mathcal{O} \setminus \mathcal{D}_1) \cup \mathcal{B} \cup \{\bigwedge_{p \in P} p\} \models C(w)$. In the course of the update, $\mathcal{D}_1$ is deleted and $\mathcal{D}_2$ used as the root of a new tree. An edge labeled with ax_3 is created and diagnosis $\mathcal{D}_3 := [ax_1, ax_4, ax_5]$ is generated. After the answer to the second query is added to the positive test cases, $\mathcal{D}_3$ is invalidated and all outgoing edge labels ax_3, ax_4 of the root $\mathcal{D}_2$ of the new tree are conflict sets for the current DPI $\langle \mathcal{O}, \mathcal{B}, \{D(v), A \sqsubseteq C\}, \{E(w), C(w)\}\rangle$, i.e. all leaf nodes are labeled by $\times$ and the tree construction is complete. So, $\mathcal{D}_2$ is returned as its probability is 1.

4 Evaluation

We evaluated our approach DIR (based on INV-QX and INV-HS-TREE) versus the standard technique STD [20] (based on QX and HS-TREE) using a set of ontologies

$$\begin{tikzcd} {[ax_2, ax_3]} \arrow[d, "ax_2"'] \\ {[ax_3, ax_4]} \end{tikzcd} \Bigg\rangle$$

D:
$\mathcal{D}_1 = [ax_2, ax_3]$
$\mathcal{D}_2 = [ax_3, ax_4]$
Query: $C(w)$
Answer: *no*

$$\Bigg\rangle \begin{tikzcd} {[ax_3, ax_4]} \arrow[d, "ax_3"] \\ {[ax_1, ax_4, ax_5]} \end{tikzcd} \Bigg\rangle$$

D:
$\mathcal{D}_2 = [ax_3, ax_4]$
$\mathcal{D}_3 = [ax_1, ax_4, ax_5]$
Query: $A \sqsubseteq C$
Answer: *yes*

$\rightarrow$ No further minimal diagnoses, return $[ax_3, ax_4]$

Fig. 1. Identification of the target diagnosis $[ax_3, ax_4]$ using interactive direct diagnosis.

created by automatic matching systems. Given two ontologies $\mathcal{O}_i$ and $\mathcal{O}_j$, a matching system outputs an *alignment* M_{ij} which is a set of *mappings* (correspondences) between semantically related entities of $\mathcal{O}_i$ and $\mathcal{O}_j$. Let $E(\mathcal{O})$ denote the set of all elements of $\mathcal{O}$ for which mappings can be produced, i.e. names of concepts. Each mapping is a tuple $\langle x_i, x_j, r, v \rangle$, where $x_i \in E(\mathcal{O}_i)$, $x_j \in E(\mathcal{O}_j)$ and x_i, x_j are either two concepts or two roles, $r \in \{\sqsubseteq, \equiv, \sqsupseteq\}$ and $v \in [0, 1]$ is a confidence value. The latter expresses the probability of a mapping to be correct. Each $\langle x_i, x_j, r, v \rangle \in M_{ij}$ can be translated to the axiom of the form $x_i \ r \ x_j$. Let $\mathcal{O}(M_{ij})$ be the set of axioms for the alignment M_{ij}, then the result of the matching process is an aligned ontology $\mathcal{O}_{ij} = \mathcal{O}_i \cup \mathcal{O}(M_{ij}) \cup \mathcal{O}_j$. The ontologies considered in this section were created by ontology matching systems participating in the Ontology Alignment Evaluation Initiative (OAEI) 2011 [5]. Each matching experiment in the framework of OAEI represents a scenario in which a user obtains an alignment M_{ij} by means of some (semi)automatic tool for two real-world *ontologies* $\mathcal{O}_i$ and $\mathcal{O}_j$.

The goal of the first experiment was to compare the performance of sequential STD and sequential DIR on a set of large, but diagnostically uncomplicated ontologies, generated for the Anatomy experiment of OAEI[1]. In this experiment the matching systems had to find mappings between two ontologies describing the human and the mouse anatomy. $\mathcal{O}_1$ (`Human`) and $\mathcal{O}_2$ (`Mouse`) include 11545 and 4838 axioms respectively, whereas the size of the alignment M_{12} produced by different matchers varies between 1147 and 1461 mappings. Seven matching systems produced a consistent but incoherent output. One system generated a consistent and coherent aligned ontology. However, this system employes a built-in heuristic diagnosis engine which does not guarantee to produce diagnoses. I.e. some axioms are removed without reason. Four systems produced ontologies which could not be processed by current reasoning systems (e.g. HermiT [16]) since consistency of these ontologies could not be checked within 2 hours.

For testing the performance of our system we have to define the correct output of sequential diagnosis which we call the target diagnosis $\mathcal{D}_t$. We assume that the only available knowledge is M_{ij} together with $\mathcal{O}_i$ and $\mathcal{O}_j$. In order to measure the performance of the matching systems the organizers of OAEI provided a *golden standard* alignment M_t considered as correct. Similarly to OAEI evaluation, in our experiments M_t was unavailable explicitly, e.g. in form of test cases, during a debugging session, just as none of matching systems has any knowledge of M_t during the competition. However, we assumed that an oracle answers debugging queries using its knowledge

[1] All ontologies and source code of programs used in the evaluation can be downloaded from http://code.google.com/p/rmbd/wiki/DirectDiagnosis. The tests were performed on Core i7, 64GB RAM running Ubuntu, Java 7 and HermiT as DL reasoner.

of M_t. Therefore, for every alignment $\mathcal{O}_{ij}$ we selected a diagnosis as target diagnosis $\mathcal{D}_t$ which is outside the golden standard, i.e. $M_t \cap \mathcal{D}_t = \emptyset$. Moreover, we used $\mathcal{O}_{ij} \setminus \mathcal{D}_t$ to answer the queries instead of M_t to ensure that the system finds exactly $\mathcal{D}_t$ and not some other diagnosis. By this procedure we mimic cases where additional information can be acquired such that no mapping of the golden standard is removed in order to establish coherence. We stress that this setting is unfavorable for diagnosis, since providing more information as test cases using the golden standard would reduce the number of queries to ask.

In particular, the selection of a target diagnosis for each $\mathcal{O}_{ij}$ output by a matching system was done in two steps: (i) compute the set of all diagnoses $\mathbf{AD}$ w.r.t. the mappings which are not in the golden standard, i.e. $\mathcal{O}(M_{ij} \setminus M_t)$, and use $\mathcal{O}_i \cup \mathcal{O}_j \cup \mathcal{O}(M_{ij} \cap M_t)$ as background theory. The set of test cases are empty. That is, the DPI is $\langle \mathcal{O}(M_{ij} \setminus M_t), \mathcal{O}_i \cup \mathcal{O}_j \cup \mathcal{O}(M_{ij} \cap M_t), \emptyset, \emptyset \rangle$. (ii) select $\mathcal{D}_t$ randomly from $\mathbf{AD}$. The prior fault probabilities of mapping axioms $ax \in \mathcal{O}(M_{ij})$ were set to $1 - v_{ax}$ where v_{ax} is the confidence value provided by the matching system.

The tests were performed for the mentioned seven incoherent alignments where the input DPI is $\langle \mathcal{O}(M_{ij}), \mathcal{O}_i \cup \mathcal{O}_j, \emptyset, \emptyset \rangle$ and the output is a diagnosis. We tested DIR and STD with both query selection strategies SPLIT-IN-HALF (SPL) and ENTROPY (ENT) in order to evaluate the quality of fault probabilities based on confidence values. Moreover, for generating a query the number of leading diagnoses was limited to $m = 9$.

The results of the first experiment are presented in Table 1. DIR computed $\mathcal{D}_t$ within 36 sec. on average and slightly outperformed STD which required 36.7 sec. The number of asked queries was equal for both methods in all but two cases resulting from ontologies produced by the `MapSSS` system. For these ontologies DIR required one query more using ENT and one query less using SPL. In general, the results obtained for the Anatomy case show that DIR and STD have similar performance in both runtime and number of queries. Both DIR and STD identified the target diagnosis. Moreover, the confidence values provided by the matching systems appeared to be a good estimate for fault probabilities. Thus, in many cases ENT was able to find $\mathcal{D}_t$ using one query only, whereas SPL used 4 queries on average. In the first experiment the identification of the target diagnosis by sequential STD required the computation of 19 minimal conflicts on average. Moreover, the average size of a minimum cardinality diagnosis over all ontologies in this experiment was 7. In the second experiment (see below), where STD is not applicable, the cardinality of the target diagnosis is significantly higher.

The second experiment was performed on ontologies of the OAEI Conference benchmark which turned out to be problematic for STD. For these ontologies we observed that the minimum cardinality diagnoses comprise 18 elements on average. In 11 of the 13 ontologies of the second experiment (see Table 2) STD was unable to find any diagnosis within 2 hours. In the other two cases STD succeeded to find one diagnosis for `csa-conference-ekaw` and nine for `ldoa-conference-confof`. However, DIR even succeeded to find 30 diagnoses for each ontology within time acceptable for interactive diagnosis settings. Moreover, on average DIR was able to find 1 diagnosis in 8.9 sec., 9 diagnoses in 40.83 sec. and 30 diagnoses in 107.61 sec. (see Column 2 of Table 2). This result shows that DIR is a stable and practically applicable method even in cases where an ontology comprises high-cardinality faults.

		HS-TREE			INV-HS-TREE		
System	Scoring	Time	#Queries	Reaction	Time	#Queries	Reaction
AgrMaker	ENT	19.62	1	19.10	20.83	1	18.23
AgrMaker	SPL	36.04	4	8.76	36.03	4	8.28
GOMMA-bk	ENT	18.34	1	18.07	14.47	1	12.68
GOMMA-bk	SPL	18.95	3	6.15	19.51	3	5.91
GOMMA-nobk	ENT	18.26	1	17.98	14.26	1	12.49
GOMMA-nobk	SPL	18.74	3	6.08	19.47	3	5.89
Lily	ENT	78.54	1	77.71	82.52	1	72.83
Lily	SPL	82.94	4	20.23	115.24	4	26.93
LogMap	ENT	6.60	1	6.30	13.41	1	11.36
LogMap	SPL	6.61	2	3.17	15.13	2	6.82
LogMapLt	ENT	14.85	1	14.54	12.89	1	11.34
LogMapLt	SPL	15.59	3	5.05	17.45	3	5.29
MapSSS	ENT	81.06	4	19.86	56.17	3	17.32
MapSSS	SPL	88.32	5	17.26	77.59	6	12.43

Table 1. HS-TREE and INV-HS-TREE applied to Anatomy benchmark. Time is given in sec, **Scoring** stands for query selection strategy, **Reaction** is the average reaction time between queries.

In the Conference experiment we first selected the target diagnosis $\mathcal{D}_t$ for each $\mathcal{O}_{ij}$ just as it was done in the described Anatomy case. Next, we evaluated the performance of sequential DIR using both query selection methods. The results of the experiment presented in Table 2 show that DIR found $\mathcal{D}_t$ for each ontology. On average DIR solved the problems more efficiently using ENT than SPL because also in the Conference case the confidence values provided a reasonable estimation of axiom fault probabilities. Only in three cases ENT required more queries than SPL. Moreover, the experiments show that the efficiency of debugging methods depends highly on the runtime of the underlying reasoner. For instance, in the hardest case consistency checking took 93.4% of the total time whereas all other operations – including construction of the search tree, generation and selection of queries – took only 6.6% of time. Consequently, sequential DIR requires only a small fraction of computation effort. Runtime improvements can be achieved by advances in reasoning algorithms or the reduction of the number of consistency checks. Currently DIR requires $O(m * |\mathcal{D}| \log(\frac{|\mathcal{O}|}{|\mathcal{D}|}))$ checks to find m leading diagnoses. A further source for improvements can be observed for the `ldoa-ekaw-iasted` ontology where both methods asked the same number of queries. In this case, ENT required only half of the consistency checks SPL did, but an average consistency check of ENT took almost twice as long as an average one for SPL. The analysis of this ontology showed that there is a small subset of axioms (hot spot) which made reasoning considerably harder. Identification of such hot spots [8] could result in a significant improvement of diagnosis runtime, since a hot spot can be resolved by suitable queries. This can be observed in the `ldoa-ekaw-iasted` case where SPL acquired appropriate test cases early and thereby found $\mathcal{D}_t$ faster.

In order to speed up the computation of conflicts and diagnoses we tested two popular module extraction methods [9, 3]. In ontology debugging these methods are used as preprocessors allowing to generate a smallest possible ontology $\mathcal{O}' \subseteq \mathcal{O}$ for a faulty

Ontology (Expressivity)	30 Diag	min $\|\mathbf{D}\|$	Scoring	Time	#Queries	Reaction	#CC	CC
ldoa-conference-confof	48.06	16	ENT	11.6	6	1.5	430	0.003
$\mathcal{SHIN}(\mathcal{D})$			SPL	11.3	7	1.6	365	0.004
ldoa-cmt-ekaw	42.28	12	ENT	48.6	21	2.2	603	0.016
$\mathcal{SHIN}(\mathcal{D})$			SPL	139.1	49	2.8	609	0.054
mappso-confof-ekaw	55.66	10	ENT	10	5	1.9	341	0.007
$\mathcal{SHIN}(\mathcal{D})$			SPL	31.6	13	2.3	392	0.021
optima-conference-ekaw	62.13	19	ENT	16.8	5	2.6	553	0.008
$\mathcal{SHIN}(\mathcal{D})$			SPL	16.1	8	1.9	343	0.012
optima-confof-ekaw	44.52	16	ENT	24	20	1.1	313	0.014
$\mathcal{SHIN}(\mathcal{D})$			SPL	17.6	10	1.7	501	0.006
ldoa-conference-ekaw	56.98	16	ENT	56.7	35	1.5	253	0.053
$\mathcal{SHIN}(\mathcal{D})$			SPL	25.5	9	2.7	411	0.016
csa-conference-ekaw	62.82	17	ENT	6.7	2	2.8	499	0.003
$\mathcal{SHIN}(\mathcal{D})$			SPL	22.7	8	2.7	345	0.02
mappso-conference-ekaw	70.46	19	ENT	27.5	13	1.9	274	0.028
$\mathcal{SHIN}(\mathcal{D})$			SPL	71	16	4.2	519	0.041
ldoa-cmt-edas	15.47	16	ENT	24.7	22	1	303	0.008
$\mathcal{ALCOIN}(\mathcal{D})$			SPL	11.2	7	1.4	455	0.002
csa-conference-edas	39.74	26	ENT	18.4	6	2.7	419	0.005
$\mathcal{ALCHOIN}(\mathcal{D})$			SPL	240.8	37	6.3	859	0.036
csa-edas-iasted	377.36	20	ENT	1744.6	3	349.2	1021	1.3
$\mathcal{ALCOIN}(\mathcal{D})$			SPL	7751.9	8	795.5	577	11.5
ldoa-ekaw-iasted	229.72	13	ENT	23871.5	9	1886	287	72.6
$\mathcal{SHIN}(\mathcal{D})$			SPL	20449	9	2100.1	517	37.2
mappso-edas-iasted	293.74	27	ENT	18400.3	5	2028.3	723	17.8
$\mathcal{ALCOIN}(\mathcal{D})$			SPL	159299	11	13116.6	698	213.2

Table 2. Interactive debugging with direct computation of diagnoses. **30 Diag** the time required to find 30 diagnoses, **min** $|\mathcal{D}|$ the cardinality of a minimum cardinality diagnosis, **Scoring** query selection strategy, **Reaction** average system reaction time between queries, **#CC** number of consistency checks, **CC** gives average time needed for one consistency check. Time is given in sec.

ontology $\mathcal{O}$ such that $\mathcal{O}'$ comprises all axioms relevant to a fault and often $|\mathcal{O}'| \ll |\mathcal{O}|$. The algorithm based on syntactic locality [9] was used in all STD tests since it improved performance of conflict set computation. The second module extraction algorithm [3] was not as effective as [9] when applied to compute conflict sets for most of our test cases. In fact, [3] tended to generate large modules including all axioms relevant to top classes of the hierarchy since all these classes are declared to be pairwise disjoint in all but two Conf ontologies. The same was observed for Anat where all top classes of the `Human` ontology are defined to be disjoint as well.

5 Conclusions

In this paper we presented a sequential diagnosis method for faulty ontologies which is based on the direct computation of diagnoses. We reduce the number of consistency checks by avoiding the computation of minimized conflict sets and by computing *some* set of diagnoses instead of a set of most probable diagnoses or a set of minimum cardinality diagnoses. The evaluation results presented in the paper indicate that the performance of the suggested sequential diagnosis system is either comparable with or outperforms the existing approach in terms of runtime and the number of queries in case a ontology includes a large number of faults. The scalability of the algorithms was demonstrated on a set of large ontologies including thousands of axioms.

References

1. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications (2007)
2. Baader, F., Knechtel, M., Peñaloza, R.: Context-dependent views to axioms and consequences of Semantic Web ontologies. J. Web Semant. 12-13, 22–40
3. Du, J., Qi, G., Pan, J.Z., Shen, Y.D.: A Decomposition-Based Approach to OWL DL Ontology Diagnosis. In: ICTAI. pp. 659–664 (2011)
4. Eiter, T., Gottlob, G.: The complexity of logic-based abduction. JACM 42(1), 1–49 (1995)
5. Euzenat, J., Ferrara, A., van Hage, W.R., Hollink, L., Meilicke, C., Nikolov, A., Ritze, D., Scharffe, F., Shvaiko, P., Stuckenschmidt, H., Sváb-Zamazal, O., dos Santos, C.T.: Final results of the Ontology Alignment Evaluation Initiative 2011. In: OM Workshop. pp. 1–29. (2011)
6. Felfernig, A., Schubert, M., Zehentner, C.: An efficient diagnosis algorithm for inconsistent constraint sets. AI EDAM 26(1), 53–62 (2012)
7. Friedrich, G., Shchekotykhin, K.: A General Diagnosis Method for Ontologies. In: ISWC. pp. 232–246. (2005)
8. Goncalves, R.S., Parsia, B., Sattler, U.: Performance Heterogeneity and Approximate Reasoning in Description Logic Ontologies. In: ISWC. pp. 82–98 (2012)
9. Grau, B.C., Horrocks, I., Kazakov, Y., Sattler, U.: Modular Reuse of Ontologies : Theory and Practice. J. Artif. Intell. Res. 31, 273–318 (2008)
10. Horridge, M., Parsia, B., Sattler, U.: Laconic and Precise Justifications in OWL. In: ISWC. pp. 323–338 (2008)
11. Horridge, M., Parsia, B., Sattler, U.: Extracting justifications from BioPortal ontologies. In: ISWC 2012. pp. 287–299 (2012)
12. Jiménez-Ruiz, E., Grau, B.C.: LogMap: Logic-based and scalable ontology matching. In: ISWC. pp. 273–288 (2011)
13. Junker, U.: QUICKXPLAIN: Preferred Explanations and Relaxations for Over-Constrained Problems. In: AAAI. pp. 167–172 (2004)
14. Kalyanpur, A., Parsia, B., Horridge, M., Sirin, E.: Finding all Justifications of OWL DL Entailments. In: ISWC. pp. 267–280 (2007)
15. de Kleer, J., Williams, B.C.: Diagnosing multiple faults. Artif. Intell. 32(1), 97–130 (1987)
16. Motik, B., Shearer, R., Horrocks, I.: Hypertableau Reasoning for Description Logics. J. Artif. Intell. Res. 36(1), 165–228 (2009)
17. Peñaloza, R., Sertkaya, B.: On the complexity of axiom pinpointing in the EL family of description logics. In: KR'10. pp. 280–289 (2010)
18. Reiter, R.: A Theory of Diagnosis from First Principles. Artif. Intell. 32(1), 57–95 (1987)
19. Satoh, K., Uno, T.: Enumerating Minimally Revised Specifications Using Dualization. In: JSAI Workshops. pp. 182–189 (2005)
20. Shchekotykhin, K., Friedrich, G., Fleiss, P., Rodler, P.: Interactive ontology debugging : two query strategies for efficient fault localization. J. Web Semant. 12-13, 88–103 (2012)
21. Stern, R., Kalech, M., Feldman, A., Provan, G.: Exploring the Duality in Conflict-Directed Model-Based Diagnosis. In: AAAI. pp. 828–834 (2012)