A direct approach to sequential diagnosis of high cardinality faults in knowledge-bases

Kostyantyn Shchekotykhin and Gerhard Friedrich and Patrick Rodler and Philipp Fleiss
1 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 probing 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 knowledge-bases. 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

Model-based diagnosis (MBD) [1] is a general method which can be used to find errors in hardware, software, knowledge-bases (KBs), orchestrated web-services, configurations, etc. In particular, ontology (KB) debugging tools [2; 3; 4] can localize a (potential) fault by finding sets of axioms \( D \subseteq KB \) called diagnoses for the KB \( KB \). Diagnoses are generated using minimal conflict sets, i.e. irreducible sets of axioms \( CS \subseteq KB \) that violate some requirements, by using a consistency checker (black-box approach). At least all axioms of a minimal diagnosis must be modified or deleted in order to formulate a fault-free knowledge-base \( KB^* \). A knowledge-base \( KB \) is faulty if some requirements, such as consistency of \( KB \), presence or absence of specific entailments, are violated.

Sequential MBD methods [5] applied to KB debugging acquire additional information in order to discriminate between diagnoses [6]. Generated queries are answered by some oracle providing additional observations about the entailments of a valid KB. As various applications show, the standard methods work very satisfactorily for cases where the number of faults (minimal conflict sets) is low (single digit number), consistency checking is fast (single digit number of seconds), and sufficient possibilities for observations are available.

However, there are situations when KBs comprise a large number of faults. For example, in ontology matching scenarios two KBs with several thousands of axioms are merged into a single one. High quality matchers (e.g. [7]) require the diagnosis of such substantially extended KBs, but could not apply standard diagnosis methods because of the large number of minimal diagnoses and their high cardinality. E.g. there are cases when the minimum cardinality of diagnoses is greater than 20.

In order to deal with hard diagnosis instances, we propose to relax the requirement for sequential diagnosis to compute a set of preferred minimal diagnoses, such as a set of most probable diagnoses. Instead, we compute some set of minimal diagnoses which can be used for query generation. This allows to use direct computation of diagnoses [8] without computing conflict sets. The direct approach was applied for non-interactive diagnosis of ontologies [9; 10] and constraints [11]. A recent approach [12] does not generate the standard HS-TREE, but still depends on the minimization of conflict sets, i.e. \(|D|\) minimized conflicts have to be discovered. Consequently, if \(|D| \gg n\), substantially more consistency checks are required, where \(|D|\) is the cardinality of the minimal diagnosis and \(n\) is the number of minimal diagnoses required for query generation.

Since we are replacing the set of most probable diagnoses by just a set of minimal 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 diagnosis problems of the ontology alignment competition [13]. 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 method is able to locate faults in all cases correctly. Moreover, for the hardest cases (i.e., more than 4 minutes overall debugging time), the additional computation costs introduced by the direct method apart from the costs needed for theorem proving are less than 50%, i.e. reasoning costs amount to more than two thirds of overall computation time.

The paper is organized as follows: Section 2 gives a brief
2 Basic concepts

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

Diagnosis of KBs. Given a knowledge-base $KB$ which is a set of logical sentences (axioms), the user can specify particular requirements during the knowledge-engineering process. The most basic requirement is satisfiability, i.e. a logical model exists. A further frequently employed requirement is coherence. Coherence requires that there exists a model s.t. the interpretation of every unary predicate is non-empty. In other words, if we add $\exists Y a(Y)$ to $KB$ for every unary predicate $a$, then the resulting KB must be satisfiable. In addition, as it is common practice in software engineering, the knowledge-engineer (user for short) may specify test cases. Test cases are axioms which must (not) be entailed by a valid KB.

Definition 1. Given a set of axioms $P$ (called positive test cases) and a set of axioms $N$ (called negative test cases), a knowledge-base $KB^*$ is valid iff it fulfills the following requirements:

1. $KB^*$ is satisfiable (and coherent if required)
2. $KB^* \models p \forall p \in P$
3. $KB^* \not\models n \forall n \in N$

Let us assume that there is a non-valid KB $KB$, then a set of axioms $D \subseteq KB$ must be removed and possibly some axioms $EX$ must be added by the user s.t. an updated $KB^*$ becomes valid, i.e. $KB^* := (KB \setminus D) \cup EX$. The goal of diagnosis is to provide information to the users which are the sets of axioms $D$, then the resulting KB must be consistent. In order to prevent unnecessary changes, $D$ is often required to be subset-minimal, i.e. the set should be as small as possible. Furthermore, we allow the user to define a set of axioms $B$ (called the background theory) which must not be changed (i.e. the correct axioms). More formally:

Definition 2. Given a diagnosis problem instance (DPI) specified by $⟨KB, B, P, N⟩$ where $KB$ is a knowledge-base, $B$ a background theory, $P$ a set of axioms which must be implied by a valid knowledge-base $KB^*$, and $N$ a set of axioms which must not be implied by $KB^*$.

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

1. $(KB \setminus D) \cup B \cup EX$ is consistent
2. $(KB \setminus D) \cup B \cup EX \models p$ for all $p \in P$
3. $(KB \setminus D) \cup B \cup EX \not\models n$ for all $n \in N$

$D$ is a minimal diagnosis iff there is no $D' \subset D$ such that $D'$ is a diagnosis. $D$ is a minimum cardinality diagnosis iff there is no diagnosis $D'$ such that $|D'| < |D|$.

The following proposition of [6] characterizes diagnoses by replacing $EX$ with the positive test cases.

Corollary 1. Given a DPI $⟨KB, B, P, N⟩$, a set of axioms $D \subseteq KB$ is a diagnosis iff

$$(KB \setminus D) \cup B \cup \{ \bigwedge_{p \in P} p \}$$

is satisfiable (coherent) and

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

Hereafter we assume that a diagnosis always exists.

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

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

Definition 3. Given a DPI $⟨KB, B, P, N⟩$, a set of axioms $CS \subseteq KB$ is a conflict set iff $CS \cup B \cup \{ \bigwedge_{p \in P} p \}$ is inconsistent (incoherent) or there is a $n \in N$ s.t. $CS \cup B \cup \{ \bigwedge_{p \in P} p \} \models n$.

Minimal conflict sets can be used to compute the set of minimal diagnoses as it is shown in [1]. The idea is that each diagnosis should include at least one element of each minimal conflict set.

Proposition 2. $D$ is a (minimal) diagnosis for the DPI $⟨KB, B, P, N⟩$ iff $D$ is a (minimal) hitting set for the set of all minimal conflict sets of the instance.

For the generation of a minimal conflict set, diagnosis systems use a divide-and-conquer method (e.g. QUICKXPLAIN [15], for short QX). In the worst case, QX requires $O(|CS| \log_{|KB|}(\frac{|KB|}{|CS|}))$ calls to the reasoner, where $CS$ is the returned minimal conflict set.

The computation of minimal diagnoses in KB debugging systems is implemented using Reiter's Hitting Set HS-TREE algorithm [11]. 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 $\vee$ (no conflicts) or $\times$ (pruned).

Each ($\vee$) node corresponds to a minimal diagnosis. The 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 of the tree generation [1]. Moreover, because of the breadth-first strategy the minimal 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 minimal diagnoses first.

Diagnoses discrimination. For many real-world DPs, a diagnosis system can return a large number of (minimal) diagnoses. Each minimal diagnosis corresponds to a different set of axioms that must be changed in order to formulate a valid $KB^*$. The user may extend the test cases $P$ and $N$ such that diagnoses are eliminated, thus identifying exactly those axioms that must be changed. For discriminating between diagnoses we assume that the user knows some of the sentences a valid $KB^*$ must (not) entail, that is the user serves as an oracle.

In this paper we consider only minimal conflict sets to avoid the issues concerning the pruning rule [1] described in [14].
Property 1. Given a DPI \( \langle KB, B, P, N \rangle \), a set of diagnoses \( D \), and a logical sentence \( Q \) representing the oracle query \( KB^+ \models Q \). If the oracle gives the answer yes then \( D_i \in D \) is a diagnosis for \( \langle KB, B, P \cup \{ Q \} \rangle \) iff both conditions hold:

\[
(KB \setminus D_i) \cup B \cup \{ \bigwedge_{p \in P} p \} \cup \{ Q \} \text{ is consistent}
\]

\[
\forall n \in N : (KB \setminus D_i) \cup B \cup \{ \bigwedge_{p \in P} p \} \cup \{ Q \} \not\models n
\]

If the oracle gives the answer no then \( D_i \in D \) is a diagnosis for \( \langle KB, B, P, N \cup \{ Q \} \rangle \) iff both conditions hold:

\[
(KB \setminus D_i) \cup B \cup \{ \bigwedge_{p \in P} p \} \text{ is consistent}
\]

\[
\forall n \in (N \cup \{ Q \}) : (KB \setminus D_i) \cup B \cup \{ \bigwedge_{p \in P} p \} \not\models n
\]

However, many different queries might exist for some set of diagnoses \( |D| > 2 \), in the extreme case exponentially many (in \( |D| \)). To select the best query, the authors in [6] suggest two query selection strategies: SPLIT-IN-HALF (SPL) and ENTROPHY (ENT). The first strategy is a greedy approach preferring queries which allow to remove half of the diagnoses in \( D \), 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 ENT measure can be obtained from statistics of previous diagnosis sessions. For instance, if the user has problems to apply the \( \exists \) operator, then the diagnosis logs are likely to contain more errors of axioms including this quantifier. Consequently, the prior fault probabilities of axioms including the \( \exists \) operator should be higher. Given the fault probabilities of axioms, one can calculate prior fault probabilities of diagnoses as well as evaluate ENT (see [6] 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 knowledge-base \( KB \) and interpreting unary predicates as classes (resp. concepts), the classification generates the inheritance (subsumption) tree, i.e. the entailments \( KB \models \forall X p(X) \rightarrow q(X) \), if \( p \) is a subclass of \( q \). Realization computes, for each individual name \( t \) occurring in a knowledge-base \( KB \), a set of most specific classes \( p \). For a valid DPI according to Definition 2. Namely, the algorithm requires: a) the background theory together with the positive and negative test cases to be consistent (Proposition 1); and b) the knowledge base to be non-valid (Definition 1). In both cases INV-QX calls VERIFY function that tests whether the set \( D \) is a minimal diagnosis or not according to Corollary 1. This function (line 17) requires a reasoner that implements consistency checking (isCONSISTENT) and allows to decide whether a set of axioms \( KB^0 \) entails some axiom \( n \) or not (ENTAILS). FindDIAG (line 8) is the main function of algorithm which takes six arguments as input. The values of the arguments \( B, KB \) and \( N \) remain constant during the recursion and are required only for the verification of requirements. Whereas the values of \( D, \Delta \) and \( KB_\Delta \) are used to provide
Algorithm 1: INV-QX(KB, B, P, N, <)

Input: KB set of faulty axioms, B set of background axioms, 
P set of positive test cases, N set of negative test cases, < ranking heuristic 
Output: a minimal diagnosis D
1 KB ← KB \ B; 
2 B' ← B \ P ; 
3 KB' ← sort(KB', <) ; 
4 if ¬VERIFY(B', ∅, ∅, N) then 
5 return 'inconsistent requirements'; 
6 if VERIFY(B', ∅, ∅, N) then return 'consistent'; 
7 return FINDDIAG(B', D, KBΔ, KB, N);

function FINDDIAG(B, D, KBΔ, KB, N) returns a minimal diagnosis D
9 if ∆ ≠ ∅ ∧ VERIFY(B, D, KB, N) then return ∅ ; 
10 if |KBΔ| = 1 then return KBΔ ; 
11 k ← SPLIT(KBΔ) ; 
12 KB1 ← GETELEMENTS(KBΔ, 1, k) ; 
13 KB2 ← GETELEMENTS(KBΔ, k + 1, |KBΔ|) ; 
14 D2 ← FINDDIAG(B, D \ KB1, KB1, KB, N) ; 
15 D1 ← FINDDIAG(B, D \ KB2, KB2, KB, N) ; 
16 return D1 \ D2 ;

function VERIFY(B, D, KB, N) returns true or false 
18 KB' ← (KB \ D) \ B ; 
19 if ¬isCONSISTENT(KB') then return false ; 
20 foreach n ∈ N do 
21 if entails(KB', n) then return false ; 
22 return true ;

a set of axioms corresponding to the actual diagnosis and two sub-DPIs provided on the next level of the recursion. The sub-DPIs are constructed by splitting KB of a given DPI (SPLIT in line 11). In the most of the implementations SPLIT simply partitions the axioms of KB into two sets of equal cardinality. The algorithm continues to divide diagnosis problems (lines 12 and 13) until it identifies that the set D is a diagnosis (line 9). In further iterations the algorithm minimizes the diagnosis by splitting it into sub-diagnoses of the form D = D' \ KBΔ, where KBΔ contains only one axiom. In case D is a diagnosis and D' is not, the algorithm decides that KBΔ is a subset of the sought minimal diagnosis. Just as the original algorithm, INV-QX always terminates and returns a minimal diagnosis for a given DPI.

INV-QX requires O(|D| log (|KBΔ|)) calls to a reasoner to find a minimal diagnosis D. Moreover, in opposite to SAT or CSP methods, e.g. [17], INV-QX can be used to compute diagnoses in cases when satisfiability checking is beyond NP. For instance, reasoning for most of KBs used in Section 4 is EXPTime-complete.

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 algorithm.

INV-HS-TREE – Construction. The algorithm is inverse to the HS-TREE algorithm in the sense that nodes are now labeled by minimal 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 minimal diagnosis D or × (pruned) which indicates that the node is closed. For each s ∈ 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 minimal 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. (redundant): D ∈ D if D ∩ H(nd) = ∅, add for each s ∈ D a node to the LIFO-queue, or
2. (pruned): × if INV-QX(KB \ H(nd), B \ H(nd), P, N) = no-diagnosis-exists, (according to Proposition 1), or
3. (compute): D if INV-QX(KB \ H(nd), B \ H(nd), P, N) = D; add D to D and add for each s ∈ D a node to the LIFO-queue.

Reuse of known diagnoses in Step 1 and the addition of H(nd) to the background theory B in Step 2 allows the algorithm to: (i) force INV-QX to search for a minimal diagnosis that is different to all diagnoses in D. Finally, if neither Step 1 nor Step 2 are applicable INV-HS-TREE calls INV-QX to compute a new minimal diagnosis D which is then added to the set D. The depth-first search strategy maintains only a set of leading diagnoses s.t. |D| ≤ m. 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 minimal diagnoses is linear and amounts to O(mk), whereas the worst case space complexity of HS-TREE is O(|C|max ? d) where |C|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 minimal 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 fault 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 minimal diagnoses are used to build a new tree. To this end, the root is labeled by any of these k minimal diagnoses. A tree is constructed as described above where the k diagnoses are incorporated for the reuse check. Note that 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 the following KB:

\[ ax_1 : \forall x \; C(x) \rightarrow A(x). \]
\[ ax_2 : \forall x \; B(x) \rightarrow C(x). \]
\[ ax_4 : \forall x \; B(x) \rightarrow C(x). \]
\[ ax_5 : \forall x \; B(x) \rightarrow \neg D(x). \]

the background knowledge \( B = \{ A(w), B(w), C(s) \} \), one positive \( P = \{ D(v) \} \) and one negative \( N = \{ E(w) \} \) test case.
Figure 1: INV-QX recursion tree. Each node shows values of FINDDIAG input variables as well as the result of VERIFY function called in line 9.

D = ∅ \quad \Delta = ∅ \quad \text{return } \{a_{x3}, a_{x2}\}

D = \{a_{x1}, a_{x2}\} \quad \Delta = \{a_{x1}, a_{x2}\} \quad \text{VERIFY } \checkmark

D = \{a_{x3}, a_{x4}, a_{x5}\} \quad \text{VERIFY } \checkmark

\text{ Query: } KB\{a_{x3}\}\cup ∅ \quad \text{Answer: no}

\rightarrow \text{No further minimal diagnoses, return } [a_{x3}, a_{x4}]

D = \{a_{x3}\} \quad \Delta = \{a_{x3}\} \quad \text{VERIFY } \checkmark

D = \{a_{x1}\} \quad \Delta = \{a_{x1}\} \quad \text{VERIFY } \checkmark

D = \{a_{x3}, a_{x1}\} \quad \Delta = \{a_{x3}\} \quad \text{VERIFY } \checkmark

D = \{a_{x3}, a_{x2}\}\cup ∅ \quad \text{VERIFY } \checkmark

Figure 2: Identification of the target diagnosis \([a_{x3}, a_{x4}]\) using INV-HS-TREE.

Let us first show how a minimal diagnosis is computed by INV-QX (see Figure 1). The algorithm starts with an empty diagnosis \(D = \emptyset\) and \(KB_D\) containing all axioms of the KB \(\overline{1}\). VERIFY returns false since \(B \cup (KB \setminus B)\) is inconsistent (line 18). Therefore, the algorithm splits \(KB_D\) into \([a_{x1}, a_{x2}]\) and \([a_{x3}, a_{x4}, a_{x5}]\) (line 11) and passes the sub-problem (line 14) to the next level of recursion \(\overline{2}\).

Since the set \(D = \{a_{x1}, a_{x2}\}\) is not a diagnosis, i.e. the ontology \(B \cup (KB' \setminus D)\) is inconsistent (line 18), the problem in \(KB_D\) is split one more time (line 11). On the second level of recursion \(\overline{3}\) the set \(D\) is a diagnosis, although not minimal. The function VERIFY returns true and the algorithm starts to analyze the found diagnosis. Therefore, it verifies whether the last extension of the set \(D\) is a subset of a minimal diagnosis \(\overline{4}\). Since the extension includes only one axiom \(a_{x3}\) and the extended set \([a_{x1}, a_{x2}]\) is not a diagnosis, the algorithm concludes that \(a_{x3}\) must be an element of the a minimal diagnosis. The leftmost branch of the recursion tree terminates and returns \([a_{x3}]\). This axiom is added to the set \(D\) and the algorithm starts investigating whether the two axioms \([a_{x1}, a_{x2}]\) also belong to a minimal diagnosis \(\overline{5}\). First, it tests the set \([a_{x3}, a_{x1}]\) \(\overline{6}\), which is not a diagnosis, and in the next iteration it identifies \([a_{x3}, a_{x2}]\) as a minimal diagnosis in node \(\overline{7}\) which is the final output of INV-QX.

In general, for the sample DPI there are four minimal diagnoses \(\{D_1\} = \{a_{x2}, a_{x3}\}, D_2 = \{a_{x3}, a_{x4}\}, D_3 = \{a_{x3}, a_{x4}, a_{x5}\}\) and four minimal conflict sets \(CS_1 = \{a_{x1}, a_{x3}\}, CS_2 = \{a_{x2}, a_{x4}\}, CS_3 = \{a_{x3}, a_{x5}\}, CS_4 = \{a_{x3}, a_{x4}\}\).

Now we show how INV-HS-TREE can be applied to find the diagnosis allowing formulation of a valid KB. Assume 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 minimal diagnosis \(D_1 := \{a_{x2}, a_{x3}\} = INV-QX(KB, B, P, N)\) to label the root node, see Figure 2. Next, it generates one successor node that is linked with the root by an edge labeled with \(a_{x2}\). For this node INV-QX(\(KB\setminus\{a_{x2}\}\), \(B \cup \{a_{x2}\}\), \(P, N\)) yields a minimal diagnosis \(D_2 := \{a_{x3}, a_{x4}\}\) disjoint with \(\{a_{x2}\}\). Now \(|D| = 2\) and a query is generated and answered as in Figure 2. Adding \(C(w)\) to the negative test cases invalidates \(D_1\) since \((KB\setminus D_1) \cup B \models C(w)\). In the course of the update, \(D_1\) is deleted and \(D_2\) used as the root of a new tree. An edge labeled with \(a_{x3}\) is created and diagnosis \(D_3 := \{a_{x1}, a_{x3}\}\) is generated. After the answer to the second query is added to the positive test cases, \(D_3\) is invalidated and all outgoing edge labels \(a_{x3}, a_{x4}\) of the root \(D_2\) of the new tree are conflict sets for the current DPI \((KB, B, \{D(v)\}, \forall X A(X) \rightarrow C(X))\), \((E(u), C(w))\), i.e. all leaf nodes are labeled by \(\times\) and the tree construction is complete. So, \(D_2\) is returned as its probability is 1.

Finally, let us compare the performance of HS-TREE [1] with the one of INV-HS-TREE. Applied to our sample DPI, the standard interactive diagnosis process using HS-TREE first calls QX [15] which returns a minimal conflict set \([a_{x1}, a_{x3}]\) (Figure 3). This minimal conflict set is used to label the root node of the HS-TREE. By reuse (R) of already computed minimal conflict sets or further calls (C) to QX (if there is no conflict set to reuse) the algorithm extends the HS-TREE until \(n = 2\) leading minimal diagnoses.
The KBs considered in this section were created by ontology matching systems participating in the Ontology Alignment Evaluation Initiative (OAEI) 2011 [13]. 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)automated tool for two real-world ontologies $KB_i$ and $KB_j$. The latter are KBs expressed by the Web Ontology Language (OWL) [18] whose semantics is compatible with the SROIQ description logic (DL). This DL is a decidable fragment of first-order logic for which a number of effective reasoning methods exist [16]. Note that, SROIQ is a member of a broad family of DL knowledge representation languages. All DL KBs considered in this evaluation are expressible in SROIQ.

The goal of the first experiment was to compare the performance of STD and DIR on a set of large, but diagnostic-almost uncomplicated KBs, generated for the Anatomy experiment of OAEI2. In this experiment the matching systems had to find correspondences between two KBs describing the human and the mouse anatomy. $KB_1$ (Human) and $KB_2$ (Mouse) include 11545 and 4838 axioms respectively, whereas the size of the alignment $M_{12}$ produced by different matchers varies between 1147 and 1461 correspondences. Seven matching systems produced a classifiable but incoherent output. One system generated a classifiable and coherent aligned KB. However, this system employs a built-in heuristic diagnosis engine which does not guarantee to produce minimal diagnoses. I.e. some axioms are removed without reason. Four systems produced KBs which could not be processed by current reasoning systems (e.g. HermiT) since these KBs could not be classified within 2 hours.

For testing the performance of our system we have to define the correct output of sequential diagnosis which we call

---

4 Evaluation

We evaluated our approach DIR (based on INV-QX and Inv-HS-TREE) versus the standard technique STD [6] (based on QX and HS-TREE) using a set of KBs created by automatic matching systems. Given two knowledge bases $KB_i$ and $KB_j$, a matching system outputs alignment $M_{ij}$, which is a set of correspondences between semantically related entities of $KB_i$ and $KB_j$. Let $Q(KB)$ denote the set of all elements of $KB$ for which correspondences can be produced, i.e. names of predicates. Each correspondence is a tuple $(x_i, x_j, r, v)$, where $x_i \in Q(KB_i)$, $x_j \in Q(KB_j)$ and $x_i$, $x_j$ have the same arity, $r \in \{\sim, \rightarrow \}$ is a logical operator and $v \in [0, 1]$ is a confidence value. The latter expresses the probability of a correspondence to be correct. Let $X$ be a vector of distinct logical variables with a length equal to the arity of $x_i$, then each $(x_i, x_j, r, v) \in M_{ij}$ is translated to the logical sentence $\forall X . x_i(X) \; r \; x_j(X)$. Let $KB(M_{ij})$ be set of axioms for the alignment $M_{ij}$, then the result of the matching process is an aligned $KB_{ij} = KB_i \cup KB(M_{ij}) \cup KB_j$. 

The KBs considered in this section were created by ontology matching systems participating in the Ontology Alignment Evaluation Initiative (OAEI) 2011 [13]. 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)automated tool for two real-world ontologies $KB_i$ and $KB_j$. The latter are KBs expressed by the Web Ontology Language (OWL) [18] whose semantics is compatible with the SROIQ description logic (DL). This DL is a decidable fragment of first-order logic for which a number of effective reasoning methods exist [16]. Note that, SROIQ is a member of a broad family of DL knowledge representation languages. All DL KBs considered in this evaluation are expressible in SROIQ.

The goal of the first experiment was to compare the performance of STD and DIR on a set of large, but diagnostic-almost uncomplicated KBs, generated for the Anatomy experiment of OAEI2. In this experiment the matching systems had to find correspondences between two KBs describing the human and the mouse anatomy. $KB_1$ (Human) and $KB_2$ (Mouse) include 11545 and 4838 axioms respectively, whereas the size of the alignment $M_{12}$ produced by different matchers varies between 1147 and 1461 correspondences. Seven matching systems produced a classifiable but incoherent output. One system generated a classifiable and coherent aligned KB. However, this system employs a built-in heuristic diagnosis engine which does not guarantee to produce minimal diagnoses. I.e. some axioms are removed without reason. Four systems produced KBs which could not be processed by current reasoning systems (e.g. HermiT) since these KBs could not be classified within 2 hours.

For testing the performance of our system we have to define the correct output of sequential diagnosis which we call

---

Footnote

[2] All KBs 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.
the target diagnosis $D_t$. We assume that the only available knowledge is $M_{ij}$ together with $\text{KB}_i$ and $\text{KB}_j$. In order to measure the performance of the matching systems the organizers of OAEI provided a golden standard alignment $M_i$ considered as correct. Nevertheless, we cannot assume that $M_i$ is explicitly available since the matching system would have used this information. W.r.t. the knowledge available, any minimal diagnosis of $\text{KB}(M_{ij})$ with $\text{KB}_i \cup \text{KB}_j$ as background theory can be selected as $D_t$. However, for every alignment we selected a minimal diagnosis as target diagnosis $D_t$ which is outside the golden standard. By this procedure we mimic cases where additional information can be acquired such that no correspondence of the golden standard is removed in order to establish coherence. We stress that this setting is unfavorable for diagnosis, since providing more information by exploiting the golden standard would reduce the number of queries to ask. Consequently, we limit the knowledge to $\text{KB}_{ij}$ and use $\text{KB}_{ij} \setminus D_t$ to answer the queries.

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

The tests were performed for the mentioned seven incoherent alignments where the input DPI is $(\text{KB}(M_{ij}), \text{KB}_i \cup \text{KB}_j, \emptyset, \emptyset)$ and the output is a minimal 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 $D_t$ 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 KBs produced by the MapSSS system. For these KBs 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 $D_t$ 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 KBs 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 KBs of the OAEI Conference benchmark which turned out to be problematic for STD. For these KBs we observed that the minimum cardinality diagnoses comprise 18 elements on average. In 11 of the 13 KBs 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 minimal diagnosis for csa-conference-ekaw and nine for ldaa-conference-conf0. However, DIR even succeeded to find 30 minimal diagnoses for each KB within time acceptable for interactive diagnosis settings. Moreover, on average DIR was able to find 1 minimal diagnosis in 8.9 sec., 9 minimal diagnoses in 40.83 sec. and 30 minimal 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 a knowledge base comprises high-cardinality faults.

In the Conference experiment we first selected the target diagnosis $D_t$ for each $\text{KB}_{ij}$ 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 $D_t$ for each KB. On average DIR solved the problems

<table>
<thead>
<tr>
<th>System</th>
<th>Scoring</th>
<th>HS-TREE</th>
<th></th>
<th>INV-HS-TREE</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time</td>
<td>#Queries</td>
<td>Reaction</td>
<td>Time</td>
<td>#Queries</td>
</tr>
<tr>
<td>AgrMaker</td>
<td>19.62</td>
<td>1</td>
<td>19.10</td>
<td>20.83</td>
<td>1</td>
</tr>
<tr>
<td>AgrMaker</td>
<td>36.04</td>
<td>4</td>
<td>8.76</td>
<td>36.03</td>
<td>4</td>
</tr>
<tr>
<td>GOMMA-bk</td>
<td>18.34</td>
<td>1</td>
<td>18.07</td>
<td>14.47</td>
<td>1</td>
</tr>
<tr>
<td>GOMMA-bk</td>
<td>18.95</td>
<td>3</td>
<td>6.15</td>
<td>19.51</td>
<td>3</td>
</tr>
<tr>
<td>GOMMA-nobk</td>
<td>18.26</td>
<td>1</td>
<td>17.98</td>
<td>14.26</td>
<td>1</td>
</tr>
<tr>
<td>GOMMA-nobk</td>
<td>18.74</td>
<td>3</td>
<td>6.08</td>
<td>19.47</td>
<td>3</td>
</tr>
<tr>
<td>Lily</td>
<td>78.54</td>
<td>1</td>
<td>77.71</td>
<td>82.52</td>
<td>1</td>
</tr>
<tr>
<td>Lily</td>
<td>82.94</td>
<td>4</td>
<td>20.23</td>
<td>115.24</td>
<td>4</td>
</tr>
<tr>
<td>LogMap</td>
<td>6.60</td>
<td>1</td>
<td>6.30</td>
<td>13.41</td>
<td>1</td>
</tr>
<tr>
<td>LogMap</td>
<td>6.61</td>
<td>2</td>
<td>3.17</td>
<td>15.13</td>
<td>2</td>
</tr>
<tr>
<td>LogMapLt</td>
<td>14.85</td>
<td>1</td>
<td>14.54</td>
<td>12.89</td>
<td>1</td>
</tr>
<tr>
<td>LogMapLt</td>
<td>15.59</td>
<td>3</td>
<td>5.05</td>
<td>17.45</td>
<td>3</td>
</tr>
<tr>
<td>MapSSS</td>
<td>81.06</td>
<td>4</td>
<td>19.86</td>
<td>56.17</td>
<td>3</td>
</tr>
<tr>
<td>MapSSS</td>
<td>88.32</td>
<td>5</td>
<td>17.26</td>
<td>77.59</td>
<td>6</td>
</tr>
</tbody>
</table>

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 system reaction time between queries.
Table 2: Sequential diagnosis using direct computation of diagnoses. 30 Diag is the time required to find 30 minimal diagnoses, min |D| is the cardinality of a minimum cardinality diagnosis, Scoring indicates the query selection strategy, Reaction is the 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.

<table>
<thead>
<tr>
<th>Ontology (Expressivity)</th>
<th>30 Diag</th>
<th>min</th>
<th>6%</th>
<th>Scoring</th>
<th>Time</th>
<th>#Queries</th>
<th>Reaction</th>
<th>#CC</th>
<th>CC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ldoa-conference-confoc</td>
<td>48.06</td>
<td>16</td>
<td>6%</td>
<td>ENT</td>
<td>11.6</td>
<td>6</td>
<td>1.5</td>
<td>430</td>
<td>0.003</td>
</tr>
<tr>
<td>SHN(D)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ldoa-cmt-ekaw</td>
<td>42.28</td>
<td>12</td>
<td>6%</td>
<td>ENT</td>
<td>11.3</td>
<td>7</td>
<td>1.6</td>
<td>365</td>
<td>0.004</td>
</tr>
<tr>
<td>SHN(D)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mappso-confok-ekaw</td>
<td>55.66</td>
<td>10</td>
<td>6%</td>
<td>ENT</td>
<td>11.1</td>
<td>5</td>
<td>1.9</td>
<td>341</td>
<td>0.007</td>
</tr>
<tr>
<td>SHN(D)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>optima-conference-ekaw</td>
<td>62.13</td>
<td>19</td>
<td>6%</td>
<td>ENT</td>
<td>11.8</td>
<td>5</td>
<td>2.6</td>
<td>553</td>
<td>0.008</td>
</tr>
<tr>
<td>SHN(D)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>optima-confok-ekaw</td>
<td>44.52</td>
<td>16</td>
<td>6%</td>
<td>ENT</td>
<td>11.6</td>
<td>5</td>
<td>1.9</td>
<td>341</td>
<td>0.007</td>
</tr>
<tr>
<td>SHN(D)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ldoa-conference-ekaw</td>
<td>56.98</td>
<td>16</td>
<td>6%</td>
<td>ENT</td>
<td>11.7</td>
<td>5</td>
<td>2.6</td>
<td>553</td>
<td>0.008</td>
</tr>
<tr>
<td>SHN(D)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>csa-conference-ekaw</td>
<td>62.82</td>
<td>17</td>
<td>6%</td>
<td>ENT</td>
<td>11.8</td>
<td>5</td>
<td>2.6</td>
<td>553</td>
<td>0.008</td>
</tr>
<tr>
<td>SHN(D)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mappso-conference-ekaw</td>
<td>70.46</td>
<td>19</td>
<td>6%</td>
<td>ENT</td>
<td>11.6</td>
<td>5</td>
<td>2.6</td>
<td>553</td>
<td>0.008</td>
</tr>
<tr>
<td>SHN(D)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ldoa-cmt-edas</td>
<td>15.47</td>
<td>16</td>
<td>6%</td>
<td>ENT</td>
<td>11.3</td>
<td>5</td>
<td>2.6</td>
<td>553</td>
<td>0.008</td>
</tr>
<tr>
<td>ALCOIN(D)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>csa-conference-edas</td>
<td>39.74</td>
<td>26</td>
<td>6%</td>
<td>ENT</td>
<td>11.3</td>
<td>5</td>
<td>2.6</td>
<td>553</td>
<td>0.008</td>
</tr>
<tr>
<td>ALCOIN(D)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>csa-edas-iasted</td>
<td>377.36</td>
<td>20</td>
<td>6%</td>
<td>ENT</td>
<td>11.3</td>
<td>5</td>
<td>2.6</td>
<td>553</td>
<td>0.008</td>
</tr>
<tr>
<td>ALCOIN(D)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ldoa-ekaw-iasted</td>
<td>229.72</td>
<td>13</td>
<td>6%</td>
<td>ENT</td>
<td>11.3</td>
<td>5</td>
<td>2.6</td>
<td>553</td>
<td>0.008</td>
</tr>
<tr>
<td>SHN(D)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mappso-edas-iasted</td>
<td>293.74</td>
<td>27</td>
<td>6%</td>
<td>ENT</td>
<td>11.3</td>
<td>5</td>
<td>2.6</td>
<td>553</td>
<td>0.008</td>
</tr>
<tr>
<td>ALCOIN(D)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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 93.1% 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, in order to generate a query, DIR requires $O(m[D] \log(\frac{KBD}{|D|}))$ 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, a sequential diagnosis session using ENT query selection method required only half of the consistency checks SPL did. However, an average consistency check made in the session using ENT took almost twice as long as an average consistency check using SPL. The analysis of this ontology showed that there is a small subset of axioms (called “hot spot” in [19]) which made reasoning considerably harder. As practice shows, they 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 $D_x$ faster. Therefore, research and application of methods allowing fast identification of such hot spots might result in a significant improvement of diagnosis runtime.

5 Conclusions

In this paper we presented a sequential diagnosis method for faulty KBs which is based on the direct computation of minimal diagnoses. We reduce the number of consistency checks by avoiding the computation of minimized conflict sets and by computing some set of minimal 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 KB includes a large number of faults. The scalability of the algorithms was demonstrated on a set of large KBs including thousands of axioms.

Acknowledgments

We would like to thank the referees for their comments, which helped improve this paper considerably.
References


