Model-Based Diagnosis with Multiple Observations

Alexey Ignatiev[1,3] , **Antonio Morgado**[1] , **Georg Weissenbacher**[2] and **Joao Marques-Silva**[1]
[1] Faculty of Science, University of Lisbon, Portugal
[2] TU Wien, Vienna, Austria
[3] ISDCT SB RAS, Irkutsk, Russia
{aignatiev, ajmorgado, jpms}@ciencias.ulisboa.pt, georg.weissenbacher@tuwien.ac.at

Abstract

Existing automated testing frameworks require multiple observations to be jointly diagnosed with the purpose of identifying common fault locations. This is the case for example with continuous integration tools. This paper shows that existing solutions fail to compute the set of minimal diagnoses, and as a result run times can increase by orders of magnitude. The paper proposes not only solutions to correct existing algorithms, but also conditions for improving their run times. Nevertheless, the diagnosis of multiple observations raises a number of important computational challenges, which even the corrected algorithms are often unable to cope with. As a result, the paper devises a novel algorithm for diagnosing multiple observations, which is shown to enable significant performance improvements in practice.

1 Introduction

The importance of system debugging cannot be overstated, given the ever growing complexity of software, hardware and cyber-physical systems. The best-known principled approach for system debugging is based on model-based diagnosis (MBD), which has a wide range of successful practical applications. Concrete examples include type error debugging [Stuckey *et al.*, 2003], design debugging [Safarpour *et al.*, 2007], software fault localization [Jose and Majumdar, 2011], debugging of web services [Ardissono *et al.*, 2005], spreadsheet debugging [Jannach and Schmitz, 2016], axiom pinpointing in description logics [Schlobach *et al.*, 2007], and debugging of relational specifications [Torlak *et al.*, 2008], among many others. Although in some settings the focus is the computation of *diagnoses* and in others the focus is the computation of *conflicts*, it is well-known that each is a minimal hitting set of the other [Reiter, 1987].

Since the original seminal work [Reiter, 1987; de Kleer and Williams, 1987], algorithms for MBD have been the subject of a number of improvements, enabling the analysis of ever more complex systems [Huang and Darwiche, 2005; Pietersma and van Gemund, 2006; de Kleer, 2008; Siddiqi, 2011; Stern *et al.*, 2012; Nica *et al.*, 2013], and also with different fault models [Feldman *et al.*, 2009]. A recent trend is the adoption of SAT-based MBD approaches [Feldman *et al.*, 2010a; Metodi *et al.*, 2014; Marques-Silva *et al.*, 2015]. MaxSAT and MaxSMT are also applied for design debugging and software fault localization [Safarpour *et al.*, 2007; Jose and Majumdar, 2011].

In software development, the use continuous integration frameworks such as Jenkins (https://jenkins.io/) and Travis CI (https://travis-ci.org/, used by GitHub) has emerged as best practice. Among other features, these frameworks support the execution of a (possibly large) number of predefined regression tests. Failing tests (or observations) require further analysis to identify possible fault locations. Compared to the standard MBD setting, the existence of multiple failing observations can reduce the number of diagnoses but also adds complexity to the analysis, raising the question how to analyze all observations in a feasible manner. Furthermore, observations in this setting are user-supplied and determined upfront. Alternative approaches that impose dependencies among observations, e.g. sequential diagnosis [Feldman *et al.*, 2010b], are not an option.

Recent algorithms for analyzing multiple (failing) observations in software [Lamraoui and Nakajima, 2014; Lamraoui and Nakajima, 2016]—as our paper demonstrates—are not guaranteed to *only* compute *minimal* diagnoses. Besides the useless diagnoses that are produced, another downside is that this approach can lead to prohibitive run times.

Motivated by the limitations of existing algorithms, this paper builds on [Ignatiev *et al.*, 2017] and has the following main contributions. First, it provides a principled approach to the simultaneous analysis of multiple failing observations. Second, it identifies key limitations in existing algorithms that analyze multiple failing observations. Third, the paper proposes fixes to such limitations and mechanisms to improve the performance of the corrected algorithms. Nevertheless, for realistic systems, the number of possible diagnoses, and their aggregation can be unmanageable. As a result, the paper develops a novel solution, based on implicit hitting set dualization. Experimental results, obtained on well-known benchmarks, highlight the efficiency gains of the proposed approach.

The paper is organized as follows. Section 2 introduces notation and definitions used in this paper. Section 3 states the problem of diagnosing multiple failing observations, and Section 3.1 details an existing algorithm for solving this problem [Lamraoui and Nakajima, 2014; Lamraoui and Nakajima, 2016] and its limitations. Section 3.2 investigates how these limitations can be addressed, and concludes that novel algorithms are required to tackle larger problems. Section 4 presents a novel scalable algorithm that exploits hitting-set dualization. Section 5 presents experimental results for standard MBD benchmarks. Section 6 concludes the paper.

2 Preliminaries

The paper uses standard model-based diagnosis (MBD) definitions, used in Reiter's seminal work [Reiter, 1987] and most modern references [Reiter, 1987; Siddiqi, 2011; Metodi *et al.*, 2014; Nica *et al.*, 2013]. In line with other recent work, the weak fault model (WFM) is assumed throughout. A system description SD is a set of first-order sentences [Reiter, 1987]. The system components, Comps, are a set of constants, $\mathsf{Comps} = \{c_1, \ldots, c_m\}$. Given a system description SD composed of a set of components Comps, each component can be declared as *healthy* or *unhealthy*. For each component $c \in \mathsf{Comps}$, $\mathsf{Ab}(c) = 1$ if c is unhealthy; otherwise $\mathsf{Ab}(c) = 0$. As in [Feldman *et al.*, 2010a; Metodi *et al.*, 2014], SD is represented as a CNF formula:

$$\mathsf{SD} \triangleq \bigwedge_{c \in \mathsf{Comps}} (\mathsf{Ab}(c) \vee \mathcal{F}_c) \tag{1}$$

where $\mathcal{F}_c$ denotes the encoding of component c.

Observations represent deviations from the expected system behavior. An observation Obs is a finite set of first-order sentences [Reiter, 1987]. Like SD, it is assumed that the observation Obs can be encoded in CNF as a set of unit clauses.

Definition 1 (Diagnosis Problem) *A system with description* SD *is faulty if it is inconsistent with a given observation* Obs *when all components are declared healthy:*

$$\mathsf{SD} \wedge \mathsf{Obs} \wedge \bigwedge_{c \in \mathsf{Comps}} \neg\mathsf{Ab}(c) \vDash \bot \tag{2}$$

The problem of diagnosis is to identify a set of components which, if declared unhealthy, restore consistency. The problem of MBD is represented by the 3-tuple $\langle\mathsf{SD}, \mathsf{Comps}, \mathsf{Obs}\rangle$.

Definition 2 (Diagnosis) *Given an MBD problem* $\langle\mathsf{SD}, \mathsf{Comps}, \mathsf{Obs}\rangle$*, the set of components* $\Delta \subseteq \mathsf{Comps}$ *is a* diagnosis *if*

$$\mathsf{SD} \wedge \mathsf{Obs} \wedge \bigwedge_{c \in \Delta} \mathsf{Ab}(c) \wedge \bigwedge_{c \in \mathsf{Comps} \setminus \Delta} \neg\mathsf{Ab}(c) \nvDash \bot \tag{3}$$

A diagnosis Δ *is* minimal *if no proper subset* $\Delta' \subsetneq \Delta$ *is a diagnosis, and* Δ *is of* minimal cardinality *if there exists no other diagnosis* $\Delta' \subseteq \mathsf{Comps}$ *with* $|\Delta'| < |\Delta|$.

In this paper, the dual of a diagnosis (often referred to as a *conflict* [Reiter, 1987]) is referred to as an *explanation*. A minimal diagnosis is a minimal hitting set of the minimal explanations, and vice-versa [Reiter, 1987].

Recent work on MBD exploited propositional encodings and Satisfiability (SAT) solvers, but also MaxSAT solvers [Safarpour *et al.*, 2007; Feldman *et al.*, 2010a; Nica *et al.*, 2013; Metodi *et al.*, 2014; Marques-Silva *et al.*, 2015]. (Each MaxSAT solution is a smallest Minimal Correction Subset (MCS). A dual concept of MCSes are Minimal Unsatisfiable Subsets (MUSes) [Birnbaum and Lozinskii, 2003; Bailey and Stuckey, 2005; Liffiton and Sakallah, 2008].) To model MBD with MaxSAT [Safarpour *et al.*, 2007; Feldman *et al.*, 2010a], SD (see (1)) represents *hard* clauses, whereas the *soft clauses* are unit clauses $(\neg\mathsf{Ab}(c))$, one for each component $c \in \mathsf{Comps}$. This is referred to as the *basic* MaxSAT encoding in this paper. Different MaxSAT solving approaches can then be applied. Alternatively, the soft clauses can be replaced by a cardinality constraint and solved iteratively with a SAT solver. Recent work on SAT-based MBD [Metodi *et al.*, 2014] develops a more sophisticated model, by using logical equivalence between the unhealthy variable of a component and its associated CNF encoding, and also by exploiting structural properties of the system description, including graph dominators and sections. Throughout the paper, the basic MaxSAT encoding of MBD is assumed [Safarpour *et al.*, 2007; Feldman *et al.*, 2010a; Marques-Silva *et al.*, 2015]. It should be observed that the MaxSAT encoding of MBD not only enables computing and enumerating minimum cardinality diagnoses, but also subset minimal diagnoses. This paper focuses on efficiently computing and enumerating minimal diagnoses in the presence of multiple (possibly many) observations.

There is a close relationship between diagnoses and minimal correction sets (MCSes), and between explanations and minimal unsatisfiable subsets (MUSes) [Reiter, 1987; Birnbaum and Lozinskii, 2003; Bailey and Stuckey, 2005]. Given the inconsistent formula (2), a minimal diagnosis Δ is such that (3) is consistent. Thus, Δ is an MCS of (2). Similarly, an explanation is a minimal hitting set of the diagnoses, and so it corresponds to an MUS of (2). As a result, enumeration of diagnoses can be obtained by enumeration of MCSes [Mencía *et al.*, 2015], and enumeration of explanations by enumeration of MUSes [Liffiton and Sakallah, 2008; Liffiton *et al.*, 2016].

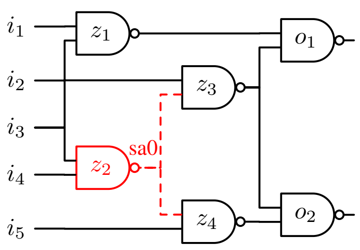
3 Diagnoses for Multiple Observations

MBD can be generalized to multiple inconsistent observations $\mathsf{Obs}_1, \ldots, \mathsf{Obs}_m$. In this setting, (3) is modified as follows for observation i, Obs_i:

$$\mathsf{SD}_i \wedge \mathsf{Obs}_i \wedge \bigwedge_{c \in \Delta} \mathsf{Ab}(c) \wedge \bigwedge_{c \in \mathsf{Comps} \setminus \Delta} \neg\mathsf{Ab}(c) \nvDash \bot \tag{4}$$

We assume that the system remains unchanged given different observations, and so SD_i is solely a *replica* of the system description SD, where the abnormal variables are shared, but the components are replicated. The distinct replica for each observation is required if all observations are analyzed jointly (as in Equation 5 below), since the actual component output values can differ for each observation.

Definition 3 (MBD with Multiple Observations) *Let* Obs_i *(*$1 \leq i \leq m$*) be a set of observations . With each observation we associate a replica of the system* SD_i*, but such that the abnormal variables are shared by the different replicas.*



(a) C17 circuit

$$
\begin{aligned}
\mathsf{Comps} &\triangleq \{z_1, z_2, z_3, z_4, o_1, o_2\} \\
\mathsf{SD} &\triangleq \textstyle\bigwedge_{c \in \mathsf{Comps}} (\mathsf{Ab}(c) \vee \mathcal{F}_c) \\
\mathcal{F}_{z_1} &\triangleq \mathsf{CNF}(z_1 \leftrightarrow \neg(i_1 \wedge i_3)) \\
\mathcal{F}_{z_2} &\triangleq \mathsf{CNF}(z_2 \leftrightarrow 0) \\
\mathcal{F}_{z_3} &\triangleq \mathsf{CNF}(z_3 \leftrightarrow \neg(i_2 \wedge z_2)) \\
\mathcal{F}_{z_4} &\triangleq \mathsf{CNF}(z_4 \leftrightarrow \neg(z_2 \wedge i_5)) \\
\mathcal{F}_{o_1} &\triangleq \mathsf{CNF}(o_1 \leftrightarrow \neg(z_1 \wedge z_3)) \\
\mathcal{F}_{o_2} &\triangleq \mathsf{CNF}(o_2 \leftrightarrow \neg(z_3 \wedge z_4))
\end{aligned}
$$

(b) MBD formulation

Observation	$\langle i_1, i_2, i_3, i_4, i_5, o_1, o_2 \rangle$
Obs_1	$\langle 0, 1, 1, 0, 1, 1, 1 \rangle$
Obs_2	$\langle 1, 1, 1, 0, 1, 1, 1 \rangle$
Obs_3	$\langle 1, 0, 0, 0, 1, 0, 1 \rangle$

(c) Example observations

Figure 1: C17 circuit with an injected stuck-at fault at gate z_2 and 3 failure revealing observations.

Obs_1:	$D_1 = \{\{z_2\}, \{z_3\}, \{z_1, z_4\}, \{z_1, o_2\}, \{z_4, o_1\}, \{o_1, o_2\}\}$
Obs_2:	$D_2 = \{\{z_2\}, \{z_3\}, \{z_4\}, \{o_2\}\}$
Obs_3:	$D_3 = \{\{z_2\}, \{z_4\}, \{o_2\}, \{z_3, o_1\}\}$

Table 1: Diagnoses for the Obs_1, Obs_2, and Obs_3 shown in Figure 1.

A minimal diagnosis $\Delta \subseteq \mathsf{Comps}$ *is a minimal set such that*

$$\bigwedge_{i=1}^{m} (\mathsf{SD}_i \wedge \mathsf{Obs}_i) \wedge \bigwedge_{c \in \Delta} \mathsf{Ab}(c) \wedge \bigwedge_{c \in \mathsf{Comps} \setminus \Delta} \neg \mathsf{Ab}(c) \nvDash \bot \qquad (5)$$

holds.

Thus, the goal is to find or enumerate subset-minimal (or cardinality-minimal) diagnoses $\Delta \subseteq \mathsf{Comps}$ that make the system consistent with *all* observations Obs_i, $1 \leq i \leq m$.

Figure 1 shows the C17 circuit from the ISCAS85 benchmark suite [Brglez and Fujiwara, 1985], which we use as a running example. Assume that a stuck-at fault injected in the circuit forces gate z_2 to output value 0 (see Figure 1a). Figure 1b shows the basic MaxSAT encoding of this faulty circuit, where the faulty gate is encoded as a constant ($z_2 \leftrightarrow 0$). Figure 1c depicts three failure revealing observations, i.e. (2) is inconsistent with each of these observations.

A conceptually simple solution to diagnose multiple observations is to consider multiple replicas of the system simultaneously, one for each observation, and then search for diagnoses that enable all replicas of the system to become consistent with their corresponding observations. MaxSAT can be used for solving the resulting problem. In practice, however, the size of the problem formulation for a non-negligible number of observations (and the associated search space) renders this approach infeasible. The next sections investigate alternative approaches aiming at scalability.

3.1 Redundant Diagnoses and the DC Algorithm

Given a set of observations $\{\mathsf{Obs}_1, \ldots, \mathsf{Obs}_m\}$, D_i denotes the set $\{\Delta_{ij}\}$ of diagnoses for observation Obs_i, with $1 \leq i \leq m$. This section investigates how diagnoses of each individual observation can be aggregated into diagnoses of the set of observations.

Definition 4 (Aggregated Diagnosis) *An* aggregated diagnosis *for the set of observations is a subset of components that contains at least one diagnosis from D_i, for $1 \leq i \leq m$, i.e. the aggregated diagnosis includes one possible diagnosis for each of the given observations.*

Observe that each aggregated diagnosis represents a sufficient condition for restoring consistency given the set of observations.

Definition 5 ((Minimal) Covering Set) *A covering set $\sigma \subseteq \mathsf{Comps}$ for the sets of diagnoses D_i, $1 \leq i \leq m$, is such that: $\forall_{1 \leq i \leq m} \exists_{\Delta_{ij} \in D_i} \,.\, \Delta_{ij} \subseteq \sigma$. A covering set is* minimal *if there is no proper subset that is also a covering set.*

As proposed in earlier work [Lamraoui and Nakajima, 2014; Lamraoui and Nakajima, 2016], the set of possible aggregated diagnoses can be obtained by computing all *covering sets* of the sets of diagnoses of each observation; however, the approach of [Lamraoui and Nakajima, 2014; Lamraoui and Nakajima, 2016] *does not guarantee* the covering sets to be minimal.

Concretely, [Lamraoui and Nakajima, 2014; Lamraoui and Nakajima, 2016] proposes the DiagCombine (DC) algorithm for computing aggregated diagnoses for a set of observations. The gist of the DC algorithm is the computation of the covering sets as described above. Each possible aggregated diagnosis is obtained by computing the union of every set of diagnoses, each associated with a different observation.

Example 1 *Let $\{\mathsf{Obs}_1, \mathsf{Obs}_2\}$ represent two observations with $D_1 = \{\{0\}, \{2\}\}$, $D_2 = \{\{0\}, \{1, 2\}\}$ denoting the sets of diagnoses, one set for each observation. The set of aggregated diagnoses for the two observations is: $\{\{0\}, \{0, 2\}, \{1, 2\}, \{0, 1, 2\}\}$. As can be observed, the aggregated diagnoses $\{0, 2\}$ and $\{0, 1, 2\}$ are not subset-minimal.*

Example 2 *The sets of diagnoses D_i, $i \in [3]$, for the faulty C17 circuit of Figure 1 are shown in Table 1. It is easy to see that diagnosis $\{z_2\}$ is a subset-minimal aggregated diagnosis. However, besides reporting it, the DC algorithm would try to combine $\{z_2\}$ with the other diagnoses of D_i, thus, producing a number of non-minimal diagnoses, e.g. $\{z_2, z_3, z_4\}$.*

As the above examples illustrate, finding covering sets of the set of diagnoses associated with each observation may produce diagnoses which are not subset-minimal.

Definition 6 (Redundant diagnosis) *An aggregated diagnosis Δ_i, that contains a diagnosis for each observation from a set of observations, is redundant if it is not subset-minimal,* *i.e. there is another aggregated diagnosis Δ_j, $i \neq j$, such that $\Delta_j \subset \Delta_i$.*

Example 3 *With respect to the diagnoses listed in Example 1, the* redundant *aggregated diagnoses are: $\{0, 2\}$, and $\{0, 1, 2\}$. Moreover, the* non-redundant *aggregated diagnoses are: $\{0\}$, and $\{1, 2\}$. Regarding the diagnoses for the faulty C17 example, plenty of* redundant *aggregated diagnoses can be seen, e.g. those combining component z_2 with the other individual diagnoses. The unit-size diagnosis $\{z_2\}$ is an example of a* non-redundant *aggregated diagnosis.*

Proposition 1 *Given a set of observations $\{\mathsf{Obs}_1, \ldots, \mathsf{Obs}_m\}$, the number of redundant diagnoses for the set of observations is in the worst-case exponential on the number of components.*

Proof. [Sketch] For each observation, the number of diagnoses is worst-case exponential on the number of components. It is simple to conceive two observations, one with a small number of diagnoses, and the other with exponentially many, such that those exponentially many diagnoses will only serve to produce redundant aggregated diagnoses given the two observations. □

Example 3 illustrates the argument used in the proof above.

The DC algorithm [Lamraoui and Nakajima, 2014; Lamraoui and Nakajima, 2016] overlooks the possibility of redundant diagnoses being computed. The number of such redundant diagnoses can be exponentially larger than the subset-minimal diagnoses.

3.2 Improvements of the DC Algorithm

A simple fix to the DC algorithm is to compute *all* covering sets of the diagnoses of each observation, aggregating each as an explanation given the observations, and then filtering the non-subset minimal diagnoses. This solution ensures that redundant diagnoses will be eliminated. Nevertheless, a potential problem with this solution is that redundant diagnoses are first generated and then discarded. Clearly, if there are exponentially many redundant diagnoses, this process can incur significant overhead. An alternative solution is to devise conditions which curb the generation of redundant diagnoses.

Proposition 2 *Suppose these exists a diagnosis Δ_t that occurs in every set of diagnoses D_i, $i = 1, \ldots, m$ associated with Obs_i. Then, Δ_t occurs in the set of aggregated diagnoses C_T, and the aggregation of Δ_t with* any *other diagnosis is redundant.*

Proof. [Sketch] Immediate by noting that the eliminated aggregated diagnoses will be proper supersets of aggregated diagnoses that are guaranteed to be computed. □

Example 4 *Revisiting Example 1, we see that given that $\{0\}$ occurs in all sets of diagnoses, there will be no other non-redundant aggregated diagnosis that also includes $\{0\}$.*

Proposition 3 *Suppose there exists a diagnosis $\Delta_t \in D_s$, such that for every set of diagnoses D_i, there exists $\Delta_{ij} \in D_i$ with $\Delta_{ij} \subseteq \Delta_t$. Then, Δ_t occurs in the set of aggregated diagnoses C_T, and the aggregation of Δ_t with* any *other diagnosis is redundant.*

Proof. [Sketch] Clearly, combining $\Delta_t \in D_s$ with every $\Delta_{ij} \in D_i$, $i \neq s$, s.t. $\Delta_{ij} \subseteq \Delta_t$, results in Δ_t being a minimal aggregated diagnosis. Hence, aggregation of Δ_t with any other diagnosis is redundant. □

Example 5 *Revisiting again Example 1, we see that $\{1, 2\}$ occurs in one set of diagnoses, and $\{2\}$ occurs in the other. Thus, $\{1, 2\}$ must occur in the aggregated set of diagnoses. Observe that the same condition can be used to explain why $\{0\}$ must occur in the aggregated set of diagnoses.*

Remark 1 *If some diagnosis appears in the aggregated set of diagnoses, then there is no need to attempt to use it for computing other aggregated diagnoses; each can be viewed as a* fixed *diagnosis.*

Example 6 *Consider the faulty C17 circuit and the sets of individual diagnoses shown in Table 1. Assuming that the target set of all non–redundant aggregated diagnoses is denoted by $\mathbb{D}$, Proposition 3 enables one to conclude that $\{\{z_2\}, \{z_1, z_4\}, \{z_1, o_2\}, \{z_3, o_1\}, \{z_4, o_1\}, \{o_1, o_2\}\} \subseteq \mathbb{D}$, without computing the covering sets as in DiagCombine. Furthermore and as discussed in Remark 1, there is no need to combine these diagnoses with any other diagnosis, i.e. they can be* dropped *from D_1, D_2, and D_3. The remaining sets of individual diagnoses yet to be combined are thus $D'_1 = \{\{z_3\}\}$, $D'_2 = \{\{z_3\}, \{z_4\}, \{o_2\}\}$, and $D'_3 = \{\{z_4\}, \{o_2\}\}$.*

Remark 2 *The previous results suggest that there are conditions under which some diagnoses behave as* absorbing elements *of the operation of aggregating diagnoses.*

Although Proposition 2 and Proposition 3 enable a significant reduction of the number of redundant aggregated diagnoses, it is also the case that the conditions are not complete, in the sense that redundant diagnoses can still be generated.

Example 7 *Let us consider a universe of three observations $\{\mathsf{Obs}_1, \mathsf{Obs}_2, \mathsf{Obs}_3\}$, with the following sets of diagnoses, one for each observation: $\{D_1 = \{\{3\}\}, D_2 = \{\{3\}, \{4\}, \{6\}\}, D_3 = \{\{4\}, \{6\}\}\}$. The set of aggregated diagnoses for the three observations is: $\{\{3, 4\}, \{3, 6\}, \{3, 4, 6\}\}$. As can be observed, the aggregated diagnosis $\{3, 4, 6\}$ is redundant. However, the conditions of Proposition 2 and Proposition 3 do not apply.*

Example 8 *A similar observation can be made with respect to the faulty C17 system. From Example 6, recall that after applying the condition of Proposition 3 to detect and then filter out a few minimal aggregated diagnoses, the sets of remaining diagnoses to be combined with use of the covering sets are $D'_1 = \{\{z_3\}\}$, $D'_2 = \{\{z_3\}, \{z_4\}, \{o_2\}\}$, and $D'_3 = \{\{z_4\}, \{o_2\}\}$. The result of the exhaustive covering set computation for D'_1, D'_2, and D'_3 is $\{\{z_3, z_4\}, \{z_3, o_2\}, \{z_3, z_4, o_2\}\}$. Observe that the last aggregated diagnosis, i.e. $\{z_3, z_4, o_2\}$, is redundant.*

While one could devise additional conditions addressing the examples above, the testing of such conditions incurs added overhead. Moreover, some other cases might not be covered by those additional conditions. The main conclusion is that it seems unrealistic to propose a closed set of conditions for filtering redundant diagnoses which runs in polynomial time.

Algorithm 1: Enumeration of minimal diagnoses

input : $\mathsf{SD}, \mathsf{Obs}_1, \ldots, \mathsf{Obs}_m$
output: $\mathbb{D} = \{\Delta_1, \Delta_2 \ldots\}$, $\mathbb{U} = \{\mathcal{U}_1, \mathcal{U}_2 \ldots\}$

1 $(\mathcal{H}_1, \ldots, \mathcal{H}_m, \mathcal{S}) \leftarrow \texttt{Encode}(\mathsf{SD}, \mathsf{Obs}_1, \ldots, \mathsf{Obs}_m)$
2 $(\mathbb{D}, \mathbb{U}) \leftarrow (\emptyset, \emptyset)$
3 **while** true**:**
4 $\quad(\mathit{st}, \Delta) \leftarrow \texttt{MinHS}(\mathbb{U}, \mathbb{D})$ *# find a min HS of* $\mathbb{U}$ *s.t.* $\mathbb{D}$
5 $\quad$**if not** *st***:**
6 $\quad\quad$**break**
7 $\quad$**foreach** $i \in \{1, \ldots, m\}$**:**
8 $\quad\quad(\mathit{st}, \kappa) \leftarrow \texttt{SAT}(\mathcal{H}_i \cup (\mathcal{S} \setminus \Delta))$
9 $\quad\quad$**if not** *st***:**
10 $\quad\quad\quad\mathcal{U} \leftarrow \texttt{Reduce}(\kappa)$ *#* $\mathcal{U}$ *is MUS of* $\mathcal{H}_i \cup (\mathcal{S} \setminus \Delta)$
11 $\quad\quad\quad\mathbb{U} \leftarrow \mathbb{U} \cup \{\mathcal{U}\}$
12 $\quad\quad\quad\texttt{ReportExpl}(\mathcal{U})$ *# report min explanation*
13 $\quad\quad\quad$**break**
14 $\quad$**else:** *# if the loop was not broken*
15 $\quad\quad\mathbb{D} \leftarrow \mathbb{D} \cup \{\Delta\}$ *# block diagnosis* Δ
16 $\quad\quad\texttt{ReportDiag}(\Delta)$ *# report min diagnosis*
17 $\quad$**foreach** $i \in \{1, \ldots, m\}$**:**
18 $\quad\quad$**if not** $\texttt{SAT}(\mathcal{H}_i \cup \mathbb{D})$**:** *# no more diagnoses exist*
19 $\quad\quad\quad$**return**
20 **return**

Furthermore, there are far more effective alternatives, which are investigated in the next section.

4 Implicit Hitting Set Dualization

This section develops an alternative algorithm which, given a (possibly large) set of observations, computes the final set of aggregated diagnoses. By construction, it filters out all redundant (non-minimal) diagnoses. Additionally, the algorithm computes (and can report) a number of explanations. In contrast with the approaches described in Section 3 and earlier work, the proposed approach is shown to scale in practice.

The proposed approach builds on recent work on hitting set dualization, which has been investigated in different contexts [Chandrasekaran *et al.*, 2011; Davies and Bacchus, 2011; Stern *et al.*, 2012; Liffiton *et al.*, 2016; Saikko *et al.*, 2016]. (However, these ideas can also be traced to the seminal work of Reiter [Reiter, 1987], and have been studied in different settings over the years, e.g. [Bailey and Stuckey, 2005; Liffiton and Sakallah, 2008] among others.)

The proposed approach is summarized in Algorithm 1. Let us denote the set of all subset-minimal aggregated diagnoses of a faulty system by $\mathbb{D}$ (analogously, the set of explanations is denoted by $\mathbb{U}$). Each Δ_i denotes a computed aggregated minimal diagnosis, and each $\mathcal{U}_j$ denotes a computed minimal explanation. Also, the basic MaxSAT encoding is assumed, i.e. given a system description SD and a list of observations $\mathsf{Obs}_1, \ldots, \mathsf{Obs}_m$, function $\texttt{Encode}()$ constructs m replicas of the CNF encoding for SD in the form of (*hard*) formulas $\mathcal{H}_1, \ldots, \mathcal{H}_m$ and a set of (*soft*) clauses $\mathcal{S}$ used for enabling/disabling the components of SD. Although the paper focuses mainly on computing subset-minimal diagnoses, the same algorithm can be used for computing cardinality-minimal diagnoses. The only difference is the implementation of function $\texttt{MinHS}()$, which can be instructed to compute either subset-minimal or cardinality-minimal hitting sets of a given set of explanations. As proposed in earlier work [Bailey and Stuckey, 2005; Stern *et al.*, 2012; Liffiton *et al.*, 2016], the algorithm iteratively computes minimal diagnoses and minimal explanations, and reports them in every iteration of the algorithm. The key objective is to find a *new* minimal hitting set of all explanations extracted so far (see line 4), at each iteration of the algorithm. (Passing $\mathbb{D}$ as an argument to $\texttt{MinHS}()$ blocks all previously computed diagnoses.) If the minimal hitting set of the explanations is not an aggregated diagnosis, i.e. it is not a diagnosis for at least one of the observations (this is checked[1] on line 8), then a new (missing) minimal explanation is extracted[2] (line 10), which is then added to the set of minimal explanations $\mathbb{U}$ (see line 11). If the computed minimal hitting set is indeed an aggregated diagnosis (for all observations), then it is discarded for future iterations by blocking the same hitting set from being computed (line 15). Observe that Algorithm 1 follows the setup of the prior instantiations of the implicit hitting set enumeration in the context of MaxSAT [Davies and Bacchus, 2011; Liffiton *et al.*, 2016; Saikko *et al.*, 2016] and its correctness relies on the known hitting set duality between MCSes and MUSes of a formula.

In contrast with other enumeration approaches proposed recently [Liffiton *et al.*, 2016], which can be viewed as targeting enumeration of explanations, Algorithm 1 will terminate as soon as all aggregated diagnoses have been computed, even if some explanations have not yet been identified (see lines 17–19). Indeed, as soon as all diagnoses for some observation have been computed and blocked, one cannot find another way to recover consistency for that observation. (Observe that this technique is standard in MCS enumeration [Mencía *et al.*, 2015; Previti *et al.*, 2018; Grégoire *et al.*, 2018] and, thus, the direct correspondence between MCSes of an unsatisfiable formula and diagnoses for a faulty system enables us to adapt the technique here.) The lines 17–19 can in practice be made optional if the goal is to compute some number K of aggregated diagnoses.

In theory, a potential drawback of Algorithm 1 is that it can compute an exponentially large number of explanations in between computed diagnoses (if the system has this many explanations). Given that every iteration of the algorithm requires an NP oracle call, this may be infeasible. However, the experimental results in Section 5 demonstrate that this worst-case scenario is not observed in practice. This is well in line with other successful uses of the implicit hitting set paradigm, where hitting set based algorithms outperformed alternative approaches and significantly pushed the state of the art [Davies and Bacchus, 2011; Ignatiev *et al.*, 2015;

[1] Here, a SAT oracle call is made w.r.t. formula $\mathcal{H}_i \cup (\mathcal{S} \setminus \Delta)$. The oracle returns a status st and an unsatisfiable core κ of the formula. Note $\mathit{st} = \mathsf{true}$ and $\kappa = \emptyset$ whenever the formula is satisfiable.

[2] Similarly to recent algorithms for MUS enumeration [Liffiton *et al.*, 2016], an off-the-shelf MUS extraction algorithm can be used in $\texttt{Reduce}()$.

Previti *et al.*, 2015; Arif *et al.*, 2015; Ignatiev *et al.*, 2016b; Liffiton *et al.*, 2016; Ignatiev *et al.*, 2016a].

5 Experimental Results

This section evaluates three approaches to MBD with multiple failing observations: the DiagCombine approach of [Lamraoui and Nakajima, 2014; Lamraoui and Nakajima, 2016], its improved version implementing the ideas of Section 3.2, and, finally, the approach based on hitting set dualization (see Section 4). The experiments[3] were performed in Ubuntu Linux on an Intel Xeon E5-2630 2.60GHz processor with 64GByte of memory. The time and memory limits for each individual instance were 1800s and 10GByte, respectively.

A prototype of the iterative hitting set dualization approach referred to as *HSD* was implemented in C++ and consists of two interacting parts. One of them computes subset-minimal or cardinality-minimal hitting sets of the set of explanations (see `MinHS`() in Algorithm 1). The other part tests consistency of the system provided that the hitting set components are disabled. (The consistency checks are done using the Glucose 3 SAT solver [Audemard *et al.*, 2013].) In the performed evaluation, HSD is configured to compute cardinality-minimal diagnoses although enumeration of subset-minimal solutions is also supported. Subset-minimal hitting sets are computed with the use of the LBX algorithm [Mencía *et al.*, 2015] for enumerating MCSes for a given unsatisfiable formula while enumeration of cardinality-minimal hitting sets is achieved with the use OLLITI/RC2 [Morgado *et al.*, 2014; Ignatiev *et al.*, 2018], the best performing MaxSAT algorithm from the MaxSAT Evaluation 2018.

Both DiagCombine and its improved version were also implemented as prototypes[4], which in the following are referred to as DC and DC*, respectively. DC implements the algorithm of [Lamraoui and Nakajima, 2014] while DC* implements the improvement discussed in Section 3.2. Both tools make use of the LBX algorithm for doing exhaustive enumeration of the individual diagnoses for each failing observation. As in HSD, Glucose 3 is used in DC and DC* as an underlying SAT solver. As discussed in Section 3.1, both DC and DC* compute a number of non-minimal diagnoses, i.e. the *redundant* diagnoses resulting from the combination step.

The test instances build on the ISCAS85 benchmark suite [Brglez and Fujiwara, 1985]. To mimic a faulty system, single stuck-at faults were injected into every gate of each ISCAS85 circuit. Assuming that each of n gates of a circuit can be stuck at either 0 or 1 results in $2n$ faulty circuits. Each of the $2n$ circuits was used to generate 100 unique observations revealing the corresponding failure. The observations were obtained by using SAT to compute a satisfying assignment for a *miter* connecting the original (correct) ISCAS85 circuit and its faulty counterpart (in which one of the gates was stuck at either 0 or 1). To illustrate the main points of the paper, the experiment targets the scenario in which multiple observations improve the quality of model-based diagnosis by reducing the number of fault candidates. Therefore, we considered only instances with at most 100 aggregated minimal diagnoses in total. We did not control the number of individual diagnoses per observation.[5] The number of benchmark instances generated in this *non-exhaustive* way is 144.

Figure 2 depicts how the considered tools compare both in terms of running time and quality of solutions reported, i.e. the number of the final aggregated diagnoses. As shown[6] in Figure 2a, HSD extensively outperforms both variants of DiagCombine. It is able to efficiently solve all 144 problem instances. The original DC solves 110 benchmarks while DC* can successfully deal with 128 instances. In terms of running time, DC and DC* are on par with each other, while HSD outperforms both of them by 2–4 orders of magnitude. In light of the downsides of DiagCombine discussed above, this result is not surprising. Figure 2b and Figure 2c confirm this intuition: Figure 2b details the number of *individual* diagnoses, which DC (and also DC*) has to compute during the first step of the DiagCombine algorithm, i.e. when enumerating diagnoses for each failing observation separately. Here, the number of individual diagnoses is compared to the number of the aggregated minimal diagnoses (computed by HSD). Recall that, by construction, each benchmark has at most 100 aggregated minimal diagnoses. This contrasts with the number of individual diagnoses that in some cases are more than 10^5, which constitutes about 5 orders of magnitude difference. The situation becomes more dramatic during the second step of the DiagCombine algorithm, i.e. after combining the diagnoses all together. This is detailed in Figure 2c. Here, the color bar of the right-hand side indicates the number of non-redundant aggregated diagnoses, ranging from 1 to 100. Thus, each point in the scatter plot shows the number of correct minimal diagnoses and the number of redundant diagnoses computed by DC and DC*. As one can observe, the improvement proposed in Section 3.2 enables DC* to significantly reduce the number of computed redundant diagnoses. Concretely, it drops by 1–6 orders of magnitude. However, in many cases DC* still computes 10-20000 non-minimal diagnoses. This together with the necessity to enumerate all individual diagnoses before their aggregation, is deemed to be a major limitation of the algorithm. In this situation, it seems natural to opt for a more efficient alternative based on the hitting set dualization.

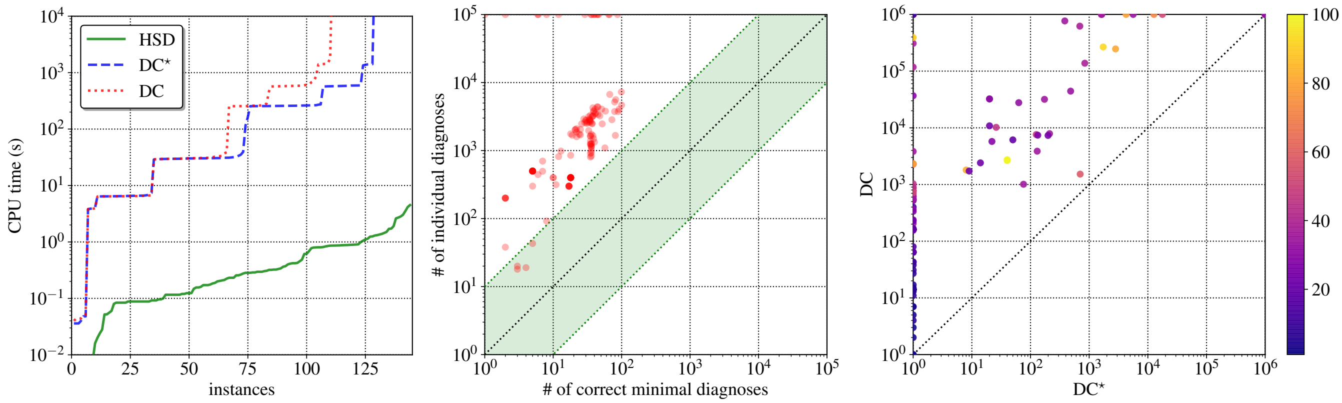
Note that the experimental evaluation targets a realistic scenario where a significant number of observations is considered simultaneously (e.g. see the Jenkins and Travis CI systems). The evaluation is conducted on a family of standard benchmarks and shows a clear advantage of the proposed hitting set based approach in practice. Nevertheless, it is

[3] The implementation of the considered approaches as well as all the benchmarks used are available online at https://github.com/alexeyignatiev/mbd-mobs.

[4] Although DiagCombine was implemented in Python, it was written on top of the PySAT toolkit [Ignatiev *et al.*, 2018], which makes use of the original low-level implementations of SAT solvers.

[5] A faulty system with its 100 observations was *discarded* if it had more than 100 minimal aggregated diagnoses, eliminating some faulty systems from the experiment. We emphasize that this filtering was done *after* the observation generation phase. However, one may come up with another range of observations for the same faulty system, which could result in a smaller number of minimal diagnoses, in which case the system could still be considered.

[6] The Y-axis is scaled logarithmically in Figure 2a.



(a) Runtime of DC, DC*, and HSD (b) Step 1 of DC/DC* (individual diags.) (c) Step 2 of DC/DC* (redundant diags.)

Figure 2: Comparison of DC, DC*, and HSD in terms of the running time and the number of diagnoses computed.

not guaranteed to significantly outperform other approaches in *all possible* practical settings. As mentioned above, if a faulty system has an exponential number of explanations, Algorithm 1 may end up enumerating them. However, the experimental results shown here do not exhibit this worst case behaviour, and on the contrary demonstrate that the existing alternative of exhaustive enumeration of individual diagnoses and their posterior aggregation is inefficient in practice.

6 Conclusions

The emergence of continuous integration frameworks motivates the need for approaches for diagnosing multiple failing observations in model-based diagnosis. As shown in this paper, existing algorithms can produce non-minimal diagnoses, in unmanageable numbers. The paper develops simple optimizations to existing algorithms. The proposed improvement is to filter non-subset-minimal aggregated diagnoses, but their number can be unwieldy. Moreover, the optimizations proposed involve conditions that prevent non-minimal diagnoses from being generated. As shown in the paper, even with these improvements, many test cases cannot be solved within a given timeout. As a result, to address the performance bottleneck of aggregating diagnoses, the paper devises a novel hitting set dualization approach. The use of hitting set dualization outperforms by orders of magnitude not only existing algorithms, but also the improvements proposed in this paper. Although the standard setting of MBD was used in the paper, the proposed ideas apply in any practical deployment of MBD, including software fault localization and design debugging, among others. This is the subject of future work.

Acknowledgements

This work was supported by FCT grants ABSOLV (PTDC/CCI-COM/28986/2017), FaultLocker (PTDC/CCI-COM/29300/2017), SAFETY (SFRH/BPD/120315/2016), and SAMPLE (CEECIND/04549/2017); the Vienna Science and Technology Fund (WWTF) through grant VRG11-005 and the Austrian Science Fund (FWF) via the Austrian National Research Network S11403-N23 (RiSE).

References

[Ardissono *et al.*, 2005] L. Ardissono, L. Console, A. Goy, G. Petrone, C. Picardi, M. Segnan, and D. T. Dupré. Cooperative model-based diagnosis of web services. In *DX*, pages 125–130, 2005.

[Arif *et al.*, 2015] M. F. Arif, C. Mencía, and J. Marques-Silva. Efficient MUS enumeration of Horn formulae with applications to axiom pinpointing. In *SAT*, pages 324–342, 2015.

[Audemard *et al.*, 2013] G. Audemard, J. Lagniez, and L. Simon. Improving Glucose for incremental SAT solving with assumptions: Application to MUS extraction. In *SAT*, pages 309–317, 2013.

[Bailey and Stuckey, 2005] J. Bailey and P. J. Stuckey. Discovery of minimal unsatisfiable subsets of constraints using hitting set dualization. In *PADL*, pages 174–186, 2005.

[Birnbaum and Lozinskii, 2003] E. Birnbaum and E. L. Lozinskii. Consistent subsets of inconsistent systems: structure and behaviour. *J. Exp. Theor. Artif. Intell.*, 15(1):25–46, 2003.

[Brglez and Fujiwara, 1985] F. Brglez and H. Fujiwara. A neutral list of 10 combinational benchmark circuits and a target translator in FORTRAN. In *ISCAS*, pages 695–698, Jun. 1985.

[Chandrasekaran *et al.*, 2011] K. Chandrasekaran, R. M. Karp, E. Moreno-Centeno, and S. Vempala. Algorithms for implicit hitting set problems. In *SODA*, pages 614–629, 2011.

[Davies and Bacchus, 2011] J. Davies and F. Bacchus. Solving MAXSAT by solving a sequence of simpler SAT instances. In *CP*, pages 225–239, 2011.

[de Kleer and Williams, 1987] J. de Kleer and B. C. Williams. Diagnosing multiple faults. *Artif. Intell.*, 32(1):97–130, 1987.

[de Kleer, 2008] J. de Kleer. An improved approach for generating max-fault min-cardinality diagnoses. In *DX*, pages 247–252, 2008.

[Feldman *et al.*, 2009] A. Feldman, G. M. Provan, and A. J. C. van Gemund. Solving strong-fault diagnostic models by model relaxation. In *IJCAI*, pages 785–790, 2009.

[Feldman *et al.*, 2010a] A. Feldman, G. Provan, J. de Kleer, S. Robert, and A. van Gemund. Solving model-based diagnosis problems with Max-SAT solvers and vice versa. In *DX*, pages 185–192, 2010.

[Feldman *et al.*, 2010b] A. Feldman, G. M. Provan, and A. J. C. van Gemund. A model-based active testing approach to sequential diagnosis. *J. Artif. Intell. Res.*, 39:301–334, 2010.

[Grégoire *et al.*, 2018] É. Grégoire, Y. Izza, and J. Lagniez. Boosting MCSes enumeration. In *IJCAI*, pages 1309–1315, 2018.

[Huang and Darwiche, 2005] J. Huang and A. Darwiche. On compiling system models for faster and more scalable diagnosis. In *AAAI*, pages 300–306, 2005.

[Ignatiev *et al.*, 2015] A. Ignatiev, A. Previti, M. H. Liffiton, and J. Marques-Silva. Smallest MUS extraction with minimal hitting set dualization. In *CP*, pages 173–182, 2015.

[Ignatiev *et al.*, 2016a] A. Ignatiev, A. Morgado, and J. Marques-Silva. Propositional abduction with implicit hitting sets. In *ECAI*, pages 1327–1335, 2016.

[Ignatiev *et al.*, 2016b] A. Ignatiev, A. Previti, and J. Marques-Silva. On finding minimum satisfying assignments. In *CP*, pages 287–297, 2016.

[Ignatiev *et al.*, 2017] A. Ignatiev, A. Morgado, and J. Marques-Silva. Model based diagnosis of multiple observations with implicit hitting sets. *CoRR*, abs/1707.01972, 2017.

[Ignatiev *et al.*, 2018] A. Ignatiev, A. Morgado, and J. Marques-Silva. PySAT: A Python toolkit for prototyping with SAT oracles. In *SAT*, pages 428–437, 2018.

[Jannach and Schmitz, 2016] D. Jannach and T. Schmitz. Model-based diagnosis of spreadsheet programs: a constraint-based debugging approach. *Autom. Softw. Eng.*, 23(1):105–144, 2016.

[Jose and Majumdar, 2011] M. Jose and R. Majumdar. Cause clue clauses: error localization using maximum satisfiability. In *PLDI*, pages 437–446, 2011.

[Lamraoui and Nakajima, 2014] S. Lamraoui and S. Nakajima. A formula-based approach for automatic fault localization of imperative programs. In *ICFEM*, pages 251–266, 2014.

[Lamraoui and Nakajima, 2016] S. Lamraoui and S. Nakajima. A formula-based approach for automatic fault localization of multi-fault programs. *JIP*, 24(1):88–98, 2016.

[Liffiton and Sakallah, 2008] M. H. Liffiton and K. A. Sakallah. Algorithms for computing minimal unsatisfiable subsets of constraints. *J. Autom. Reasoning*, 40(1):1–33, 2008.

[Liffiton *et al.*, 2016] M. H. Liffiton, A. Previti, A. Malik, and J. Marques-Silva. Fast, flexible MUS enumeration. *Constraints*, 21(2):223–250, 2016.

[Marques-Silva *et al.*, 2015] J. Marques-Silva, M. Janota, A. Ignatiev, and A. Morgado. Efficient model based diagnosis with maximum satisfiability. In *IJCAI*, pages 1966–1972, 2015.

[Mencía *et al.*, 2015] C. Mencía, A. Previti, and J. Marques-Silva. Literal-based MCS extraction. In *IJCAI*, pages 1973–1979, 2015.

[Metodi *et al.*, 2014] A. Metodi, R. Stern, M. Kalech, and M. Codish. A novel SAT-based approach to model based diagnosis. *J. Artif. Intell. Res. (JAIR)*, 51:377–411, 2014.

[Morgado *et al.*, 2014] A. Morgado, C. Dodaro, and J. Marques-Silva. Core-guided MaxSAT with soft cardinality constraints. In *CP*, pages 564–573, 2014.

[Nica *et al.*, 2013] I. Nica, I. Pill, T. Quaritsch, and F. Wotawa. The route to success - a performance comparison of diagnosis algorithms. In *IJCAI*, 2013.

[Pietersma and van Gemund, 2006] J. Pietersma and A. J. C. van Gemund. Temporal versus spatial observability in model-based diagnosis. In *ICSMC*, pages 5325–5331, 2006.

[Previti *et al.*, 2015] A. Previti, A. Ignatiev, A. Morgado, and J. Marques-Silva. Prime compilation of non-clausal formulae. In *IJCAI*, pages 1980–1988, 2015.

[Previti *et al.*, 2018] A. Previti, C. Mencía, M. Järvisalo, and J. Marques-Silva. Premise set caching for enumerating minimal correction subsets. In *AAAI*, pages 6633–6640, 2018.

[Reiter, 1987] R. Reiter. A theory of diagnosis from first principles. *Artif. Intell.*, 32(1):57–95, 1987.

[Safarpour *et al.*, 2007] S. Safarpour, H. Mangassarian, A. G. Veneris, M. H. Liffiton, and K. A. Sakallah. Improved design debugging using maximum satisfiability. In *FMCAD*, pages 13–19, 2007.

[Saikko *et al.*, 2016] P. Saikko, J. Berg, and M. Järvisalo. LMHS: A SAT-IP hybrid MaxSAT solver. In *SAT*, pages 539–546, 2016.

[Schlobach *et al.*, 2007] S. Schlobach, Z. Huang, R. Cornet, and F. van Harmelen. Debugging incoherent terminologies. *J. Autom. Reasoning*, 39(3):317–349, 2007.

[Siddiqi, 2011] S. A. Siddiqi. Computing minimum-cardinality diagnoses by model relaxation. In *IJCAI*, pages 1087–1092, 2011.

[Stern *et al.*, 2012] R. T. Stern, M. Kalech, A. Feldman, and G. M. Provan. Exploring the duality in conflict-directed model-based diagnosis. In *AAAI*, 2012.

[Stuckey *et al.*, 2003] P. J. Stuckey, M. Sulzmann, and J. Wazny. Interactive type debugging in Haskell. In *Haskell*, pages 72–83, 2003.

[Torlak *et al.*, 2008] E. Torlak, F. S. Chang, and D. Jackson. Finding minimal unsatisfiable cores of declarative specifications. In *FM*, pages 326–341, 2008.