# Exploring the Duality in Conflict-Directed Model-Based Diagnosis

**Roni Stern and Meir Kalech**
Information Systems Engineering
Ben Gurion University
Beer-Sheva, Israel 85104
roni.stern@gmail.com, kalech@bgu.ac.il

**Alexander Feldman and Gregory Provan**
Department of Computer Science
University College Cork
College Road, Cork, Ireland
a.feldman@cs.ucc.ie, g.provan@cs.ucc.ie

**Abstract**

A model-based diagnosis problem occurs when an observation is inconsistent with the assumption that the diagnosed system is not faulty. The task of a diagnosis engine is to compute diagnoses, which are assumptions on the health of components in the diagnosed system that explain the observation. In this paper, we extend Reiter's well-known theory of diagnosis by exploiting the duality of the relation between conflicts and diagnoses. This duality means that a diagnosis is a hitting set of conflicts, but a conflict is also a hitting set of diagnoses. We use this property to interleave the search for diagnoses and conflicts: a set of conflicts can guide the search for diagnosis, and the computed diagnoses can guide the search for more conflicts. We provide the formal basis for this dual conflict-diagnosis relation, and propose a novel diagnosis algorithm that exploits this duality. Experimental results show that the new algorithm is able to find a minimal cardinality diagnosis faster than the well-known Conflict-Directed A*.

Model-based diagnosis (MBD) is a field that has developed a well-defined theory and algorithms for computing multiple sets of faulty components in a malfunctioning system. In MBD, a model of the system is first built. A diagnoser then observes the system to predict its behavior by the model. Discrepancies between the observation and the prediction are used as the input for a diagnosis algorithm which produces a set of possible faults that can explain the observation.

The characterization of MBD by Reiter (1987) in terms of first-order logic has led to significant developments in the field. Reiter presents a conflict-directed algorithm as follows. Given an observation that is inconsistent with the model's predictions, the diagnosis process first identifies symptoms that represent inconsistencies (discrepancies) between the system's model (description) and the system's actual behavior. Each symptom identifies minimal conflicts, i.e., sets of conflicting components which cannot all be functioning correctly. Second, the process computes from the minimal conflicts the minimal diagnoses, which are the smallest sets of components that intersect all candidate sets. Therefore, finding the minimal diagnosis set is accomplished in two steps: first generating candidate sets from symptoms, and then calculating a minimal set of faulty components.

This approach has several drawbacks. The first concerns computational tractability: conflict-based diagnosis actually has to solve two NP-hard problems: (1) computing the minimal conflicts, and from these, (2) computing the minimal hitting sets to generate the minimal diagnoses. It is important to compute minimal conflicts and minimal diagnoses, because the number of possible conflicts and diagnoses is always exponential in the number of system components.

The second drawback concerns diagnostics completeness: most conflict-based approaches are incomplete, i.e., cannot discover all possible conflicts. Instead, they use sound but incomplete techniques, e.g., (Williams and Ragno 2007).

The General Diagnostic Engine (GDE) (de Kleer and Williams 1987)) is one of the best-known implementations of this classical approach. Several approaches have been used to address the computational problems and to optimize this approach. First, GDE used an ATMS (de Kleer 1986) to manage and optimize the conflict generation and minimization. Then, several "focusing" methods were introduced to limit the search; for example, probabilities representing failure likelihoods for the components were exploited to focus search on the most likely conflicts (de Kleer and Williams 1989). A search-based approach, conflict-directed A* (CDA*) (Williams and Ragno 2007), exploits the probabilities to implement a best first search hitting set algorithm on a conflict set that is generated incrementally.

In this article, we propose a novel algorithm for MBD that exploits key properties of the traditional conflict-based approach to diagnostics inference. We extend Reiter's theory of diagnosis by showing a dual relation between conflicts and diagnoses: not only is a diagnosis a hitting set of conflicts, but a conflict is also a hitting set of diagnoses. This duality can then be used to interleave the search for diagnoses and conflicts, i.e., we can use conflicts to guide the search for diagnoses, and use the computed diagnoses to guide the search for more conflicts. This approach is sound and complete, and does not have the overhead of maintaining all minimal conflicts, as does the ATMS.

Our contributions are as follows:

- We extend the theoretical work on the relation between conflicts and diagnoses.

- We point out the symmetry between diagnoses and conflicts, and focus on the conflict to diagnosis relation in order to design a diagnosis engine that uses both directions of this symmetry.
- We show that the resulting diagnosis engine is a sound and complete anytime diagnosis engine that finds all diagnoses, and can identify minimum-cardinality diagnoses.
- We empirically compare our dual diagnosis engine with CDA* to show that it can find minimal cardinality diagnoses in substantially more cases and for observations with larger minimal cardinality than can CDA*, e.g., finding even minimal cardinality diagnoses of size 12.

## Related Work

This section briefly reviews related methods for computing diagnoses. The best-known algorithm for MBD, GDE and its subsequent improvements, is based on using the ATMS (de Kleer 1986) to compute and manage conflicts. This approach is sound and complete with respect to identifying all minimal conflicts and diagnoses; however, it may face computational challenges, which have been remedied to some extend by extensions, such as NGDE (de Kleer 2009). NGDE was developed as a faster version of GDE, especially since it employs a better hitting set algorithm. Note that improved hitting set algorithms are orthogonal to our approach, and can used to further optimize our approach as well.

An alternative to this approach is conflict-directed A* (Williams and Ragno 2007), which uses LTMS, a TMS that is based on Boolean Constraint Propagation (BCP), to extract conflicts that will direct the search. This conflict generation is different from that of GDE in two aspects: 1) Conflict generation with LTMS is polynomial, and much faster than ATMS, 2) LTMS is not complete, meaning that it may not return all conflicts or even minimal conflicts, and 3) while GDE generates all conflicts upfront, in CDA* conflicts are generated one at a time, only when a candidate diagnosis is found to be inconsistent. In practice, CDA* has shown to be much more efficient than GDE on standard diagnosis benchmarks. Importantly, CDA* is not complete, meaning that it does not guarantee to return all the subset-minimal diagnoses. This is discussed in details in this paper.

Another incomplete, but computationally efficient approach is SAFARI, which uses stochastic local search to compute diagnoses. SAFARI does not guarantee minimal cardinality nor completeness, although it has shown to be able to diagnose far larger models than competing algorithms (Feldman, Provan, and van Gemund 2010).

A quite different approach to diagnostic inference is the use of compilation, i.e., rewriting the system description into an alternative representation that speeds inference. Two popular compilation targets are OBDDs (Torasso and Torta 2003) and DNNF (Darwiche 1998). Compilation is attractive in that inference is typically linear in the size of the compiled representation; however, there is no guarantee that the compiled representation will not be of size exponential in the number of system components.

In contrast to the Safari algorithm and the compilation-based approaches described above, our approach extends the classical method for identifying diagnoses by computing conflicts. However, we use the duality of conflicts and diagnoses to improve the inference. Hence, in comparison to GDE, diagnoses are returned without maintaining all minimal conflicts, but instead we use the computed diagnoses to guide the inference of minimal conflicts, which is in turn used to find more diagnoses. In contrast to conflict-directed A*, our approach is guaranteed to be sound and complete with respect to both the minimal conflicts and diagnoses.

## Concepts and Definitions

Our discussion continues by formalizing some MBD notions. This paper uses the traditional diagnostic definitions (de Kleer and Williams 1987), except that we use propositional logic terms (conjunctions of literals) instead of sets of failing components.

A *model* of an artifact is represented as a propositional formula over some set of variables. We discern subsets of these variables as *assumable* and *observable*.

**Definition 1** (Diagnostic Problem)**.** A diagnosis problem DP is defined as the quadruple $\langle \mathrm{SD}, \mathrm{COMPS}, \mathrm{OBS}, \alpha \rangle$, where SD is a propositional formula over a set of variables $V$, $\mathrm{COMPS} \cup \mathrm{OBS} \subseteq V$, COMPS is the set of assumables, OBS is the set of observables, and $\alpha$ is a (possibly partial) assignment of the variables in OBS.

We assume that $\mathrm{SD} \not\models \bot$, i.e., SD is not faulty (does not lead to diagnoses) when there is no observation. We also assume that $\mathrm{COMPS} \cap \mathrm{OBS} = \emptyset$.

Let $\mathrm{COMPS} = \{h_i\}$ for $i = 1, 2, \ldots, n$. We use positive assignments $h_i = \mathbf{True}$, or simply positive literals $h_i$, to denote healthy components; conversely, we use negative assignments $h_i = \mathbf{False}$, or negative literals $\neg h_i$, to denote faulty components.

**Definition 2** (Health Assignment)**.** Given a diagnosis problem DP, an assignment $h$ to **all** variables in COMPS is defined as a health assignment.

A health assignment $h$ is a conjunction of propositional literals. The set of positive literals in $h$ is denoted as $Lit^{+}(h)$ and the set of negative literals in $h$ is denoted as $Lit^{-}(h)$.

**Definition 3** (Diagnosis and Conflicts)**.** Given a diagnostic problem DP and health assignments $\omega$ and $\lambda$. $\omega$ is a diagnosis iff $\mathrm{SD} \wedge \alpha \wedge \omega$ is satisfiable, and $\lambda$ is a conflict iff $\mathrm{SD} \wedge \alpha \wedge \lambda$ is not satisfiable.

A common problem in MBD is that a given diagnosis problem may have a large amount of diagnoses. To elevate this, the MBD literature have proposed a range of types of "preferred" diagnosis. This turns the MBD problem into an optimization problem. In the following definition we consider the common subset-ordering.

**Definition 4** (Subset-Minimality)**.** A diagnosis $\omega^{\subseteq}$ is defined as subset-minimal, if no other diagnosis $\tilde{\omega}^{\subseteq}$ exists such that $Lit^{-}(\tilde{\omega}^{\subseteq}) \subset Lit^{-}(\omega^{\subseteq})$. A conflict $\lambda^{\subseteq}$ is defined as subset-minimal, if no other conflict $\tilde{\lambda}^{\subseteq}$ exists such that $Lit^{+}(\tilde{\lambda}^{\subseteq}) \subset Lit^{+}(\lambda^{\subseteq})$.

The cardinality of a diagnosis $\omega$ and a conflict $\lambda$ are denoted as $|\omega|$ and $|\lambda|$, respectively ($|Lit^{-}(\omega)| = |\omega|$ and

$|Lit^+(\lambda)| = |\lambda|$). Diagnosis cardinality gives us another partial ordering: a diagnosis is defined as *minimal cardinality* iff it minimizes the number of negative literals.

**Definition 5** (Cardinality-Minimality)**.** A diagnosis $\omega^{\leq}$ is defined as cardinality-minimal if no other diagnosis $\tilde{\omega}^{\leq}$ exists such that $|\tilde{\omega}^{\leq}| < |\omega^{\leq}|$. A conflict $\lambda^{\leq}$ is defined as cardinality-minimal if no other conflict $\tilde{\lambda}^{\leq}$ exists such that $|\tilde{\lambda}^{\leq}| < |\lambda^{\leq}|$.

A cardinality-minimal diagnosis is a subset-minimal diagnosis, but the opposite does not hold. There are subset-minimal diagnoses that are not cardinality-minimal diagnoses.

Given a diagnosis problem DP we denote the set of all diagnoses of SD $\wedge$ $\alpha$ as $\Omega$, the set of all subset-minimal diagnoses as $\Omega^{\subseteq}$, the set of all cardinality-minimal diagnoses as $\Omega^{\leq}$, the set of all conflicts as $\Lambda$, the set of all subset-minimal conflicts as $\Lambda^{\subseteq}$, and the set of all cardinality-minimal conflicts as $\Lambda^{\leq}$.

## Duality of Diagnoses and Conflicts

A *hitting set* of a set of conflicts $\Lambda$ is a set of components $C$ such that for any $\lambda \in \Lambda$ there is an element $c \in C$ such that $c \in Lit^+(\lambda)$.

**Theorem 1** ((Reiter 1987), Theorem 4.5)**.** *A health assignment $\omega^{\subseteq}$ is a subset-minimal diagnosis iff $Lit^-(\omega^{\subseteq})$ is a subset-minimal hitting set of all the subset-minimal conflicts (i.e., $\Lambda^{\subseteq}$).*

In order to use the above theorem to obtain diagnoses, one needs to obtain all the subset-minimal conflicts of the given diagnosis problem. Note that a hitting set of a sub-collection of all the subset-minimal conflicts is not guaranteed to be a diagnosis. For example, consider a diagnosis problem with two minimal conflicts $\{C_1, C_2\}$ and $\{C_3, C_4\}$. Clearly the hitting sets of only $\{C_1, C_2\}$, e.g., $\{C_1\}$ and $\{C_2\}$ are not hitting sets of all the conflicts, and thus they are not diagnoses according to Theorem 1. Furthermore, a hitting set of a set of conflicts that is not minimal (even if every minimal conflict is contained in one of these conflicts) is also not guaranteed to be a diagnosis. In the example above, a hitting set of the two conflicts $\{C_1, C_2, C_5\}$ and $\{C_3, C_4, C_5\}$, i.e., adding $C_5$ to both conflicts above, will generate $\{C_5\}$ as a hitting set of these non-minimal conflicts, which is not a diagnosis. Thus, generating consistent diagnoses with Theorem 1 requires both finding *all* conflicts, and making sure that they are minimal. This is often a challenging task. While finding all subset-minimal conflicts is potentially intractable, it is often very easy to find a partial set of not necessarily subset-minimal conflicts.

**Corollary 1** (Required condition for diagnosis)**.** *For every subset-minimal diagnosis $\omega^{\subseteq}$ it must hold that $Lit^-(\omega^{\subseteq})$ is a (not necessarily subset-minimal) hitting set of any set of conflicts.*

*Proof.* A hitting set of all subset-minimal conflicts ($\Lambda$) is a hitting set of all conflicts, since for every subset-minimal diagnosis $\omega^{\subseteq}$ we have that $Lit^-(\omega)^{\subseteq}$ is a hitting set of all the subset-minimal conflicts ($\Lambda^{\subseteq}$), and thus every subset of all conflicts as well. □

Corollary 1 is useful because it can be used to help find diagnoses even if only a partial set of (not necessarily) subset-minimal conflicts is available. This is done by directing the search for diagnoses to consider only health assignments that are hitting sets of the available set of conflicts. This observation is used by CDA$^*$ (Williams and Ragno 2007).

The first contribution of this paper is to point out the symmetry of this relation between conflicts, hitting sets and diagnoses. To this end, we define a *hitting set* of a set of diagnoses $\Omega$ in a similar manner to the way we defined a hitting set of conflicts, i.e., a hitting set of a set diagnoses $\Omega$ is a set of components $C$ such that for any $\omega \in \Omega$ there is an element $c \in C$ such that $c \in Lit^-(\omega)$.

**Theorem 2** (Necessary and Sufficient Condition for a Conflict)**.** *A health assignment $\lambda^{\subseteq}$ is a subset-minimal conflict iff $Lit^+(\lambda^{\subseteq})$ is a subset-minimal hitting set of all the subset-minimal diagnoses (i.e., $\Omega^{\subseteq}$).*

This observation is important for two reasons. First, there are many domains where finding conflicts is important. For example, in Software Fault Localization (SFL), one can compute a minimal hitting set (MHS) of a program spectra to compute diagnoses (Abreu and van Gemund 2010). If we are going to use SFL methods (ranking, etc.) for MBD we need efficient way to extract conflicts from SD $\wedge$ $\alpha$. Furthermore when searching for diagnoses directed by conflicts, one can use the previously computed diagnoses to help the search for more conflicts.

However, as explained above, finding conflicts with diagnoses requires finding all subset-minimal diagnoses to guarantee that the hitting set of these diagnoses is a conflict. Obviously, this is often a hard task. The following corollary extends Theorem 2 to overcome this problem in a similar manner to the way Corollary 1 extends Theorem 1.

**Corollary 2** (Necessary Condition for a Conflict)**.** *For every subset-minimal conflict $\lambda^{\subseteq}$ we have that $Lit^+(\lambda^{\subseteq})$ is a (not necessarily subset-minimal) hitting set of any set of diagnoses.*

The proof is analogous to the proof of Corollary 1 (the two corollaries are dual) and we omit it for brevity.

**Proposition 1** (Complement of a Diagnosis/Conflict)**.** *A health assignment $h$ is either a diagnosis or a conflict.*

While trivial, this proposition is interesting when stated in common diagnosis and conflict terms (Reiter 1987; de Kleer and Williams 1987). For a given health assignment $h$, if SD $\wedge$ $\alpha$ $\wedge$ $h$ is consistent, then the set of unhealthy components $Lit^-(h)$ is often refereed to as *the diagnosis*. Alternatively, if SD $\wedge$ $\alpha$ $\wedge$ $h$ is not consistent, then the set of healthy components $Lit^+(h)$ is often refereed to as *the conflict*. We can rephrase Proposition 1 as follows: if a set of components $\delta$ is not a diagnosis, then COMPS $\setminus$ $\delta$ is a conflict, and vice versa (Reiter 1987). This proposition is important when searching for diagnoses guided by conflicts. as is done by CDA* (as described above), or when searching for conflicts guided by diagnoses. If the hitting set of the available conflicts (diagnoses) is not a diagnosis (conflict), it can be added to the set of available conflicts (diagnoses) to further direct the following search for diagnoses (conflicts).

Next, we describe how these theoretical statements can be used to construct an efficient diagnostic algorithm.

## Conflict-Diagnosis Search

The algorithm we propose in this paper can be considered as a generalization of previous conflict-directed diagnostic engines. A diagnosis problem is a specialization of a *search problem*. A search problem consists of an initial state, a goal state and a set of state-transition operators. In the diagnosis problem, a state would be a health assignment. The initial state is a heath assignment $h$ such that $Lit^{-}(\omega) = \emptyset$, i.e., assuming that all the components are healthy. A goal state is a diagnosis (a health assignment that is consistent with the system description SD and the observation $\alpha$). Applying an operator to a state $h$ corresponds to setting a health variable $v$ that is positive in $h$ to be faulty, i.e., we "flip" the sign of $v$. The branching factor of a state $h$ is the size of $Lit^{+}(h)$.

A brute-force approach to find diagnoses can be to apply any exhaustive search on the problem space, e.g., breadth-first search. To check if a candidate health assignment is a goal (i.e., it is a diagnosis), a consistency check is needed. This approach is not feasible for non-trivial systems. To find the smallest diagnoses (i.e., a cardinality-minimal one), such an approach would have to visit at least $\binom{|COMPS|}{|\omega^{\leq}|-1)}$ where $|\omega^{\leq}|$ is the cardinality of the minimal cardinality diagnosis. For instance, in a system of $1000$ components, to find a minimal cardinality diagnosis of size $5$, a brute-force diagnosis engine must first verify the absence of a diagnosis consisting of $4$ components (there are more than $10^{11}$ such combinations).

### Building Blocks

In general, conflict-directed diagnostic algorithms (1) search for conflicts, (2) generate candidate diagnoses by considering hitting sets of the generated set of conflicts, and (3) identify which of these candidates are diagnoses by performing consistency checks. We briefly describe the basic building blocks used in the design of this kind of algorithms.

**[Conflict generation (CON):]** All subset-minimal conflicts can be generated with ATMS (de Kleer 1986), while not necessarily minimal conflicts can be generated with LTMS. Naturally, ATMS is much more computationally intensive (NP-Hard) than LTMS (polynomial).

**[Computation of hitting sets (HS):]** Finding the minimal cardinality hitting set is a well-known NP-complete problem (Garey and Johnson 1979). On the other hand, finding subset-minimal hitting set in problems that arise from weak-fault models (WFMs) can be done in time that is polynomial in the number of components times the number of sets to hit (in our case - number of conflicts).

**[Consistency checking (SAT):]** In its most general form, preforming a consistency check requires solving a SAT problem, which is also NP-complete. On the other hand, there are cases where it is enough to apply Boolean Constraint Propagation (BCP) to verify consistency, which is polynomial (this is equivalent to LTMS).

There are several non-trivial trade-offs in combining those three building blocks. If CON generates all subset-minimal conflicts, then any hitting set of these conflicts is guaranteed to be a diagnosis. Thus, there is no need to execute SAT (see Theorem 1). This approach is taken by GDE. CDA*, on the other hand, searches for hitting sets of conflicts with an A* search (which in the case where every component has the same probability of failure turns into a breadth-first search), grows the conflict set incrementally, does not guarantee minimal conflicts, and thus applies a consistency check to verify that a hitting set of the conflicts is indeed a diagnosis. Note that computing cardinality-minimal diagnoses is computationally more difficult than computing subset-minimal diagnoses.

### Primal-Dual Diagnostic Search

Similar to CDA*, one can find conflicts even without finding all diagnoses, and even with diagnoses that are not subset-minimal. This will require more effort in the hitting set (HS) and consistency check (SAT) components. Furthermore, in a weak-fault model (WFM) it is very easy to find non-subset-minimal diagnoses, as any super set of a diagnosis is also a diagnoses (de Kleer, Mackworth, and Reiter 1992).

Next, we describe how to construct a diagnosis algorithm that is built from the previously described building blocks. By changing the type of HS, SAT and MIN we can exploit various properties of the search space. We call this algorithm the Primal-Dual Diagnostic Search algorithm (PDDS). Importantly, PDDS can be used in both directions, either for computing diagnoses, or for computing conflicts.

**Algorithm 1:** Primal-Dual Diagnostic Search (PDDS)

```
   Input: DP, a diagnostic problem
   Input: $\Omega$, set of consistent hitting sets
   Input: $\Lambda$, a collection of sets to hit
   Input: ConsistencyCheck, a subroutine to
          perform consistency checking
   Input: $K$, maximal number of hitting sets
 1 for $k = 1 \ldots K$ do
 2     $\omega \leftarrow$ compute a new hitting set of $\Lambda$
 3     if no new hitting sets were found then
 4         return True
 5     if ConsistencyCheck(SD $\wedge \alpha \wedge \omega$) then
 6         $\Omega \leftarrow \Omega \cup \{\omega\}$
 7     else
 8         $\lambda \leftarrow \omega$
 9         $\Lambda \leftarrow \Lambda \cup \{$Minimize($\lambda$)$\}$
10 return False
```

Algorithm 1 computes up to $K$ hitting sets, which are consistent (or inconsistent) with some theory SD. When applied to search for a conflict, PDDS searches for hitting sets of a given set of diagnoses, and the consistency check verifies that the found hitting set is indeed a conflict (i.e., a healthy assignment that is UNSAT with the system description and observation). When applied to search for a diagnosis, PDDS searches for a hitting set of a given set of conflicts, and the consistency check verifies that the resulting hitting set is a diagnosis (i.e., the assignment is consistent with SD $\wedge$ $\alpha$). PDDS accepts as a parameter the set it needs to hit ($\Omega$) and a method for checking consistency (CONSISTENCYCHECK). A hitting set algorithm is run and a hitting set $\omega$ is returned. The only assumption we have on the hitting set algorithm used is that it is complete, i.e., it will return a hitting set that is not already in $\Omega$ if such exists.

If $\omega$ is consistent, it is added to the set of consistent hitting sets $\Omega$. Otherwise, based on Definition 3 and Proposition 1, it is actually a conflict and is added to the set of conflicts $\Lambda$. Optionally, it is possible to try to minimize the conflict (line 9 in Algorithm 1). Below we describe methods to minimize a given inconsistent hitting set, such that it still remains inconsistent. This will ensure that $\omega$ will not be returned in the subsequent iteration as a hitting set of $\Omega$. This process is repeated until no hitting set of $\Omega$ exists, in which case PDDS returns True as it has computed all subset-minimal diagnoses, or until a maximum number of $K$ iterations have been performed, in which case PDDS will return $False$.

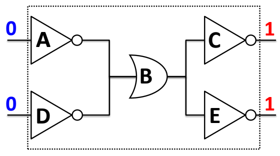

Figure 1: An example diagnosis problem.

As an example of PDDS, consider the small electrical circuit displayed in Figure 1. Assume that we are searching for diagnoses. Recall that PDDS receives as input a set of sets to hit ($\Lambda$) and a set of previously found consistent hitting sets ($\Omega$). In our example, assume that a single conflict $\lambda$ is given, where the conflict set of the components $\{A, B, C\}$ is denoted by $Lit^{-}(\lambda) = \{A, B, C\}$ and $Lit^{+}(()\lambda) = \{D, E\}$. Also, assume that and no previous diagnoses exists. In other words, $\Lambda = \{\lambda\}$ and $\Omega = \emptyset$. First, PDDS finds a hitting set of $\Lambda$. Let $A$ be the hitting set that was found, and thus the corresponding candidate diagnosis $\omega$ has $Lit^{-}(\omega) = \{A\}$ and $Lit^{+}(\omega) = \{B, C, D, E\}$. Clearly, $\omega$ is not consistent, and thus $\omega$ is added as a conflict to $\Lambda$. Next, the hitting set algorithm may find the candidate diagnosis $\lambda$ with $Lit^{-}(\omega) = \{B\}$, which hits both the original conflict ($\{A, B, C\}$) and the new conflict ($\{B, C, D, E\}$). This time, $\omega$ is consistent, and thus it is added as a diagnosis to $\Omega$. Note that in this case it is not possible to minimize this diagnosis (line 9) as $\omega$ contains only $\{B\}$. We omit the rest of the execution of PDDS due to space limitations.

## Properties of PDDS

We first describe theoretical properties for the case where PDDS is used to find diagnoses, i.e., where $\Lambda$ is a set of conflicts, and CONSISTENCYCHECK is a SAT solver. However all results hold for the dual case as well.

**Theorem 3** (Soundness and Completeness)**.** *Given* SD $\in$ **WFM** *and any set of initial conflicts* $\Lambda$*, after* $K$ *iterations, PDDS is guaranteed to return True for a sufficiently large* $K$*, and returns all subset-minimal diagnoses.*

**Proof:**(sketch) The soundness of all diagnoses is guaranteed by the consistency check in line 5. Assume by negation that PDDS is not complete. This means that there exists a diagnosis $\omega$ that is not found by PDDS when it returns true. PDDS returns true only if has found all hitting sets of $\Lambda$. This means that $\omega$ is not a hitting set of $\Lambda$, since PDDS only halts after all consistent hitting sets have been found. The set $\Lambda$ is initialized by a set of conflicts, and more conflicts may be added to it during the search. Thus, all the sets in $\Lambda$ are conflicts. According to Corollary 1 every diagnosis must hit every set of conflicts, and therefore, $\omega$ cannot be a diagnosis, resulting in a contradiction.$\square$
Note that Theorem 3 holds even if the initial set of conflicts contain non-minimal conflicts, and only a subset of all the conflicts. This is an important property since finding non-minimal conflicts is, in general, easy.

Since the number of minimal subset diagnoses is often exponentially large, it is common to prefer diagnoses of smaller cardinality. While it is possible to find all minimal subset diagnoses and then sort them according to their cardinality, this is clearly inefficient. A more efficient diagnosis algorithm returns first all the diagnoses with minimal cardinality, and in general return all the diagnoses in order of increasing cardinality. Such a diagnosis algorithm can, for example, halt when all minimal cardinality diagnosis.

PDDS can be easily converted to return all the diagnoses in order of increasing cardinality, by using a hitting set algorithm (line 2 in Algorithm 1) that returns hitting sets in order of increasing cardinality. Examples of such HS algorithms are breadth-first search and iterative deepening. In the rest of this paper we assume that PDDS uses such HS algorithms, and thus returns diagnoses in order of increasing cardinality. Thus, running PDDS with $\Omega = \emptyset$ and $K = 1$ will return a minimal cardinality diagnosis. Also, it is easy to modify PDDS to return only minimal cardinality diagnoses: simply terminate when the HS algorithm returns a HS of size

Having conflicts of small cardinality in $\Lambda$ will result in faster HS search and less iterations of PDDS. We perform the PDDS minimization of conflicts and diagnoses (line 9) by employing a greedy algorithm similar to the one used in SAFARI (Feldman, Provan, and van Gemund 2010). First we flip a random health variable in $Lit^{+}(\lambda)$, and use a consistency check to see if it is still a conflict. Note that this consistency check is the opposite of the one used to check if a hitting set is a diagnosis. If after this flip $\lambda$ is not a conflict we try to flip another health variable in $Lit^{+}(\lambda)$ and so on. This way we advance greedily from a non-minimal conflict to a subset-minimal one.

## Switching Diagnostic Engine (SDE)

The duality of conflicts and diagnoses allows PDDS to search for conflicts as well as diagnoses. Next, we describe a diagnostic algorithm, called Switching Diagnostic Engine (SDE) and outlined in Alg. 2, which finds diagnoses by interleaving two PDDS algorithms, one searching for conflicts and the other searching for diagnoses. We show that the combination of these two search directions results in a better and more versatile diagnostic algorithm. Throughout its execution, SDE maintains a set of conflicts and a set of

**Algorithm 2:** Switching Diagnostic Engine (SDE)

```
Input: DP, diagnostic problem
1 $\Lambda \leftarrow$ initial set of conflicts
2 $\Omega \leftarrow$ initial set of diagnoses (may be empty)
3 while True do
4     PDDS($\Lambda$, $\Omega$, UNSAT solver, 1)
5     if PDDS($\Omega$, $\Lambda$, SAT solver, 1) = True then Halt
```

| Iteration | HS | Cons. | Conflicts | Diagnoses |
|---|---|---|---|---|
| 0 | | | $\{A, B, C, D, E\}$ | $\{A, B, C, D, E\}$ |
| 1 Conf. | {A} | No | $\{A, B, C, D, E\}$ | $\{B, C, D, E\}$ |
| 1 Diag. | {A} | No | $\{B, C, D, E\}$ | $\{B, C, D, E\}$ |
| 2 Conf. | {B} | No | $\{B, C, D, E\}$ | $\{C, D, E\}$ |
| 2 Diag. | {B} | Yes | $\{B, C, D, E\}$ | {**B**}, $\{C, D, E\}$ |

Table 1: Several iterations of SDE.

diagnoses, denoted as $\Lambda$ and $\Omega$ respectively. The algorithm begins by finding an initial conflict and an initial diagnosis (lines 1 – 2). These initial diagnosis and conflict need not be minimal, and thus it is easy to find them. For example, the trivial health assignments of all components being faulty is a diagnosis if SD $\in$ **WFM**, and the trivial health assignment of having all the components healthy can be checked if it is a conflict or a diagnosis with a single call to SAT solver. In our experiments we used the same stochastic local search described above to minimize the initial conflict or diagnosis.

Next, we run a single iteration of PDDS by setting $K = 1$, first searching for a new conflict, and then run a single iteration of PDDS to search for a new diagnosis. Recall that when PDDS searches for a diagnosis, it may also compute conflicts. Conversely, when PDDS searches for conflicts it may also compute diagnoses. As the search progresses, more diagnoses are found, which directs the conflict search to find consistent conflicts, and this in turn directs the search for diagnoses. When no new hitting sets of conflicts exist, the search can halt, as all subset-minimal diagnoses have been found (line 5).

To demonstrate how SDE works, Table 1 lists a partial example of how SDE works on the small electrical circuit shown in Figure 1. The *iteration* column marks the current SDE iteration. The search for diagnoses (line 5) and the search for conflicts (line 4) are given different lines in the table, denoted by “Diag.” and “Conf.”, respectively. The values in the *HS* column are the hitting sets found by PDDS. When searching for diagnoses, this is a hitting set of the conflicts and vise-verca. The values in the *consistent* column state if the found hitting set was consistent (i.e., if it was a conflict/diagnoses). The *conflicts* and *diagnoses* columns show the $Lit^+()$ and $Lit^-()$ of the conflicts and diagnoses there were found so far, respectively. Several simplifications were made to SDE to allow the reader to follow this example. First, the hitting set search in PDDS was done in a breadth-first search manner, tie-breaking according to lexicographical order. Second, we did not apply the stochastic minimization process in PDDS (line 9 in Algorithm 1). It is important to emphasize that when implementing SDE, using these simplifications is not advised and will result in significant performance degradation.

SDE has several helpful properties. It is complete and sound, as it is based on PDDS which is complete and sound. Furthermore, SDE is in fact an diagnostic *anytime algorithm*. It starts with an initial diagnosis, and it continues to returns diagnoses very quickly, either by PDDS when searching for diagnoses or by PDDS when searching for conflicts. As the search progresses, more diagnoses will be returned, returning a sequence of diagnosis as required by an anytime diagnosis algorithm.

Like PDDS, SDE can also be modified to return diagnoses in order of increasing cardinality. In particular, SDE can return the minimal cardinality diagnosis. This is done by using a variant of PDDS that returns only minimal cardinality diagnosis, when SDE searches for diagnoses (line 5 in Algorithm 2). Such a variant of PDDS was described in the PDDS properties section above. In the experimental results that are described next, we used this version of SDE, that is modified to return first the minimal cardinality diagnosis.

## Experimental Results

| Alg. | 74283 | 74182 | 74181 | c432 | c499 | c880 | c1355 |
|---|---|---|---|---|---|---|---|
| #Obs | 202 | 250 | 350 | 76 | 208 | 284 | 210 |
| Minc. | 5 | 5 | 7 | 8 | 22 | 26 | 21 |
| CDA* | 100% | 100% | 47% | 33% | 9% | 5% | 0% |
| SDE | 99% | 100% | 79% | 88% | 30% | 14% | 9% |

Table 2: Instances solved under under 30 sec.

Next, we provide experimental results, comparing the dual search approach with the conflict-directed approach. Specifically, we have implemented the dual search inside the Lydia framework (Feldman, Provan, and van Gemund 2009), and compared it against CDA* which is also implemented in the same framework. As a diagnosis task, we chose the task of returning a single minimal cardinality diagnosis, and bounded the run time by 30 seconds.

Recently, several techniques have been introduces to improve the performance of diagnosis algorithms. This include an efficient hitting set algorithm (de Kleer 2011) and the use of a type of abstraction called cones (Siddiqi and Huang 2007; Siddiqi 2011). Both technique are orthogonal to SDE and CDA*, and can be applied to SDE and CDA* in a similar manner to improve their performance. Additionally, we did not compare against the SAFARI diagnosis engine (Feldman, Provan, and van Gemund 2010), since the task in these experiments was to find a minimal cardinality diagnosis, and SAFARI is not guaranteed to return such diagnoses.

Experiments where performed on the standard 74XXX benchmark and on the small circuits from the ISCAS85 benchmark. On larger circuits CDA* was not able to solve any instance. We used a partial set of the observations used by (Feldman, Provan, and van Gemund 2010).

Table 2 shows the percentage of instances where the minimal cardinality was found under the 30 second time-limit. The first row (marked “#Obs”) in the table shows the number of observations that were used in the experiments. The second line (marked “Minc.”) gives the largest minimal cardinality diagnosis found in the observations of every system. As can clearly be seen, SDE is able to find a minimal cardinality in substantially more instance. For example, SDE found a minimal cardinality diagnosis in 88% of the instances for the c432 system while CDA* was able to do the same for only 33% of the instances.

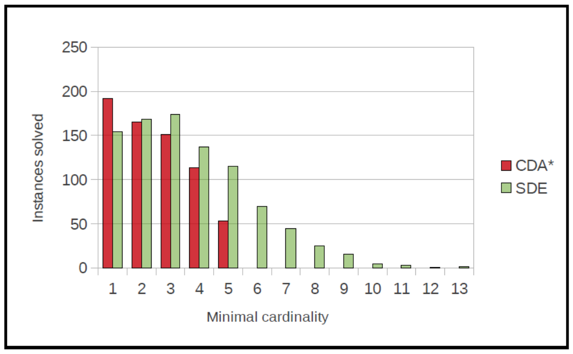

Figure 2: # instances solved, grouped by cardinality.

To identify the instances where SDE gains its advantage, consider the results in Figure 2. The y-axis shows the number of instances (over all the systems) where a minimal cardinality diagnosis is found under 30 seconds. The x-axis groups instances with the same minimal cardinality. These results show that while CDA* is limited to solving low minimal cardinality instances, SDE is able to solve instances with larger minimal cardinality.

This can be explained as follows. CDA* and SDE both generate conflicts when a candidate diagnosis was found to be inconsistent. In addition, SDE searches for conflicts (using PDDS) to find more conflicts to direct the search for diagnoses. Thus, the results in Figure 2 demonstrates the trade off between the overhead of searching for more conflicts, and the expected gain of having more conflicts to direct the search for diagnoses. That explains the performance of CDA* in finding diagnoses with very low cardinality. It does not need the redundant conflicts to direct the search to the low minimal cardinality diagnosis. On the other hand, when the cardinality of the diagnosis is high, the additional conflicts found by SDE are helpful for directing the diagnosis search towards these high cardinality.

## Conclusion and Future Work

In this paper we presented the symmetry between diagnoses and conflicts. We showed that just as diagnosis is a hitting set of conflicts, a conflict is a hitting set of diagnoses. As such, conflict-directed diagnosis algorithms can be converted into *diagnosis-directed* conflict search algorithms. This enables creating a single algorithm, named PDDS, that can find either diagnoses or conflicts. PDDS is shown to be complete, sound and can return diagnoses in order of increasing cardinality. Furthermore, we introduce the SDE diagnosis algorithm, which interleaves two PDDS searches: one for diagnoses and the other for conflicts. SDE uses conflicts to guide the search for diagnoses, and uses the computed diagnoses to guide the search for more conflicts. We empirically compared SDE with CDA* on well known benchmarks. Results showed that SDE outperforms CDA* in searching for the first minimal cardinality diagnosis. The performance difference between our algorithm, SDE, and CDA* increased with diagnosis cardinality. This is an important result since the diagnosis problem becomes harder as cardinality grows, and in this paper we presented an algorithm that can cope with diagnoses with high cardinality better than previous algorithms.

## References

Abreu, R., and van Gemund, A. J. C. 2010. Diagnosing multiple intermittent failures using maximum likelihood estimation. *Artif. Intell.* 174:1481–1497.

Darwiche, A. 1998. Model-based diagnosis using structured system descriptions. *Journal of Artificial Intelligence Research* 8:165–222.

de Kleer, J., and Williams, B. 1987. Diagnosing multiple faults. *Artificial Intelligence* 32(1):97–130.

de Kleer, J., and Williams, B. C. 1989. Diagnosis with behavioral modes. In *IJCAI*, 1324–1330.

de Kleer, J.; Mackworth, A.; and Reiter, R. 1992. Characterizing diagnoses and systems. *Artificial Intelligence* 56(2-3):197–222.

de Kleer, J. 1986. An assumption-based TMS. *Artificial Intelligence* 28(2):127–162.

de Kleer, J. 2009. Minimum cardinality candidate generation. In *Proc. DX’09*, 397–402.

de Kleer, J. 2011. Hitting set algorithms for model-based diagnosis. In *22th International Workshop on Principles of Diagnosis (DX-11)*.

Feldman, A.; Provan, G.; and van Gemund, A. 2009. The Lydia approach to combinational model-based diagnosis. In *Proc. DX’09*, 403–408.

Feldman, A.; Provan, G.; and van Gemund, A. 2010. Approximate model-based diagnosis using greedy stochastic search. *Journal of Artificial Intelligence Research* 38:371–413.

Garey, M. R., and Johnson, D. S. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman.

Reiter, R. 1987. A theory of diagnosis from first principles. *Artificial Intelligence* 32(1):57–95.

Siddiqi, S., and Huang, J. 2007. Hierarchical diagnosis of multiple faults. In *Proc. IJCAI’07*, 581–586.

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

Torasso, P., and Torta, G. 2003. Computing minimum-cardinality diagnoses using OBDDs. *KI 2003: Advances in Artificial Intelligence* 224–238.

Williams, B., and Ragno, R. 2007. Conflict-directed A* and its role in model-based embedded systems. *Journal of Discrete Applied Mathematics* 155(12):1562–1595.