SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

November 2018, Vol. 61 112101:1–112101:14
https://doi.org/10.1007/s11432-017-9273-5

Efficient zonal diagnosis with maximum satisfiability

Meng LIU[1,3], Dantong OUYANG[1,3], Shaowei CAI[2] & Liming ZHANG[1,3*]

[1]*College of Computer Science and Technology, Jilin University, Changchun* 130012*, China;*
[2]*State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing* 100049*, China;*
[3]*Key Laboratory of Symbolic Computation and Knowledge Engineering, Ministry of Education, Jilin University, Changchun* 130012*, China*

Received 11 July 2017/Revised 15 August 2017/Accepted 27 September 2017/Published online 18 May 2018

Abstract Model-based diagnosis (MBD) has been widely acknowledged as an effective diagnosis paradigm. However, for large scale circuits, it is difficult to find all cardinality-minimal diagnoses within a reasonable time. This paper proposes a novel method that takes a significant step in this direction. The idea is to divide a circuit into zones and compute the cardinality-minimal diagnoses by finding subset-minimal diagnoses with cardinality-minimal via a maximum satisfiability (MaxSAT) solver on an abstracted circuit that is composed of these zones instead of all components. We also propose a new propagate-extend method for extending the seed-TLDs to obtain all cardinality-minimal diagnoses efficiently. We implement our method with a state-of-the-art core-guided MaxSAT solver, and present evidence that it significantly improves the diagnosis efficiency on ISCAS-85 circuits. Our method outperforms SATbD, which was recently shown to outperform most complete MBD approaches using satisfiability (SAT).

Keywords model-based diagnosis, MaxSAT, subset-minimal diagnoses, cardinality-minimal diagnoses, zone, propagate-extend

Citation Liu M, Ouyang D T, Cai S W, et al. Efficient zonal diagnosis with maximum satisfiability. Sci China Inf Sci, 2018, 61(11): 112101, https://doi.org/10.1007/s11432-017-9273-5

1 Introduction

In general, diagnosing a system refers to identifying the root causes of encountered errors. Model-based diagnosis (MBD) is a well-known diagnosis paradigm. Given a system model and an observation of the system's inputs and outputs, not consistent with the expected behavior, the task of MBD is to compute a subset of the components that explain the observed behavior, according to some minimality criterion. It plays an important role in many approaches to knowledge representation and reasoning [1–7].

Previous approaches to the MBD problem can be categorized into several classes, including conflict-based [8–10], constraint-based [11–15], compilation-based [16,17], and the use of duality in conflict-based approaches [18].

Recently, the use of satisfiability (SAT) (or MaxSAT) solvers in constraint-based MBD approaches has emerged as a new paradigm that is usually referred to as the SAT-based (or MaxSAT-based) [19–22] approach to MBD. In these approaches, cardinality-minimal diagnoses are computed by iterative SAT (or MaxSAT) solvers. Given a propositional formula in conjunctive normal form (CNF), the SAT problem [23,24] is to determine whether there is an assignment to the variables such that the formula is

* Corresponding author (email: limingzhang@jlu.edu.cn)

evaluated as true, and the MaxSAT problem is to find an assignment that satisfies as many clauses as possible. In particular, in the partial MaxSAT problem, clauses are divided into hard and soft clauses, and the goal is to satisfy all the hard clauses and as many soft clauses as possible. MBD can be encoded into a partial MaxSAT problem.

Because of its high efficiency and high adaptability, MaxSAT has become a popular method of solving MBD. An algorithm for compiling the MBD problem for MaxSAT was first presented in [25]. Subsequently, several MaxSAT-based methods have been proposed for MBD. Kutsuna et al. [26] used a partial MaxSAT algorithm to solve several diagnostic automotive control problems. Similarly, Chen et al. [27] used a partial MaxSAT to debug sequential circuits. Stochastic local search MaxSAT algorithms can also be used for computing cardinality-minimal diagnoses [12, 28]. In a recent study on the MaxSAT-based MBD method, Marques-Silva et al. [29] provided a method to downscale the MBD problem via filtering the circuit using observations. However, this method only obtained one top-level diagnosis (TLD) in the experiments.

Empirical evidence shows that although MaxSAT-based solvers can compute diagnoses in many cases, their performance degrades when the circuit size or number of injected faults increases. A typical method for improving the performance of MBD solving on large circuit is the hierarchical diagnosis method [17]. In this method, the circuit is abstracted using the concept of a dominator [30]. The TLDs are obtained using HD05 [31] for the abstracted circuit. Then, the diagnosis of the abstracted subcircuit replaces the dominator in the TLDs to generate a new diagnosis. This iterative diagnosis process is very time-consuming. Nevertheless, thanks to the theory in [17], it has been shown to significantly reduce the scale of the problem and thus can deal with larger circuits. Metodi et al. [13, 14] presented the concept of a section into which a circuit is divided, where the components that have the same system outputs are in the same section. A section is used to estimate the size of the cardinality-minimal diagnosis. In terms of downscaling circuits, the approach proposed by Metodi et al. is the same as the hierarchical diagnosis method [17]. Next, the TLDs are obtained using SAT for the sections and cones (as described below), and then all cardinality-minimal diagnoses are acquired by an iterative enumeration of the replacements while consistency is checked using SAT.

However, the complexity of the iterative process is subject to combinatorial explosion and moreover, consistency checking can be computationally expensive. Hence, as mentioned above, Marques-Silva et al. [29] provided a method to further downscale the problem by filtering the circuit with observations. They also demonstrated the equivalence between the primary circuit and the filtered one.

In this paper, we propose a novel MaxSAT-based approach to MBD. First, we propose a novel method that substantially downscales the problem. The key idea is to divide a circuit into zones that are extended with seed-components. Then, the seed-TLDs are acquired for the zones instead of all the components via a MaxSAT solver. In addition, we obtain these seed-TLDs in order of cardinality. Namely, our approach obtains seed-TLDs that are cardinality-minimal diagnoses.

Second, we propose a new propagate-extend method for extending the seed-TLDs to efficiently obtain all cardinality-minimal diagnoses. Note that for previous SAT/MaxSAT-based MBD methods, after obtaining TLDs, it is necessary to call a SAT solver to obtain all the cardinality-minimal diagnoses from the TLDs. In contrast, our propagate-extend method can obtain all cardinality-minimal diagnoses from the TLDs without calling a SAT solver. The zone-minimal diagnoses are sufficient to obtain the opposite output of the expected value.

Finally, the corresponding element in the seed-TLD is replaced to obtain all cardinality-minimal diagnoses. This process is cost effective. More importantly, to obtain cardinality-minimal diagnoses, we abandon the extended diagnoses that are not cardinality minimal. Experiments show that extending 100 seed-TLDs to all the cardinality-minimal diagnoses that they represent only takes 0.1 s. Compared with the time needed to compute the seed-TLDs, this time can be ignored in most cases. In addition, we compare our approach to the state-of-the-art algorithm using SAT for MBD SATbD [13, 14]. Our approach outperforms it by one to two orders of magnitude in terms of runtime.

The paper is organized as follows. Section 2 introduces some preliminaries. Section 3 presents the zonal diagnosis (ZD) approach and the details of the main process. Section 4 compares the ZD approach

with the latest version of SATbD [13, 14], a state-of-the-art solver for computing all cardinality-minimal diagnoses. Finally, we conclude the paper in Section 5.

2 Preliminaries

The MBD problem appears when the normal behavior of a system is observed to be violated because of some faulty components in the circuit. The weak fault model is considered in this paper, which ignores the mode of abnormal behavior of the components. An MBD problem can be represented as a 3-tuple $\langle \mathrm{SD}, \mathrm{Comps}, \mathrm{Obs} \rangle$, where SD is a system description, Comps is a set of components, and Obs is a given observation. In the MBD problem, some components might be unhealthy. An unhealthy component $c \in \mathrm{Comps}$ can be stated as a unary predicate Ab such that $\mathrm{Ab}(c)$ is true when the behavior of component c is abnormal. A system description SD is defined in [32]. When c behaves correctly, denoted as propositional formula φ_c, SD can be defined as a conjunction of $(\neg \mathrm{Ab}(c) \rightarrow \varphi_c)$. In other words, each healthy component is consistent with its correct behavior. A description of the diagnosis problem and related concepts can be found in [32, 33].

Definition 1 (Diagnosis problem)**.** Given an MBD problem $\langle \mathrm{SD}, \mathrm{Comps}, \mathrm{Obs} \rangle$, the diagnosis problem appears when, under the assumption that all components are declared healthy, there is an inconsistency between the system description and the given observation, i.e., $\mathrm{SD} \cup \{\neg \mathrm{Ab}(c) \,|c \in \mathrm{Comps}\} \cup \mathrm{Obs}$ is inconsistent.

Definition 2 (Diagnosis)**.** Given an MBD problem $\langle \mathrm{SD}, \mathrm{Comps}, \mathrm{Obs} \rangle$, $\Delta \subseteq \mathrm{Comps}$ is a diagnosis when $\mathrm{SD} \cup \mathrm{Obs} \cup \{\mathrm{Ab}(c) \,|c \in \Delta\} \cup \{\neg \mathrm{Ab}(c) \,|c \in \mathrm{Comps} \setminus \Delta\}$ is consistent. In addition, Δ is a subset-minimal diagnosis when any $\Delta' \subset \Delta$ is not a diagnosis and is a cardinality-minimal diagnosis when any $|\Delta'| < |\Delta|$ is not a diagnosis.

Obviously, the cardinality-minimal concept is stricter than subset-minimal. Namely, a cardinality-minimal diagnosis is a subset-minimal diagnosis, while a subset-minimal diagnosis is not necessarily a cardinality-minimal diagnosis because its size may not minimal. In this paper, cardinality-minimal diagnoses are obtained by obtaining subset-minimal diagnosis in order of cardinality and then discarding those with sizes that are not cardinality minimal based on their definitions.

MBD can be encoded into the partial MaxSAT problem, where the system description and observation are modeled as hard clauses while the components are represented as soft clauses. Previous MaxSAT-based MBD methods focus on dividing the circuit into cones according to dominators, which are described as follows. Let O denote a special component to which every circuit output is connected. Component v is a dominator of component u if all paths from u to O include v. Component v is an immediate dominator of u if every other dominator of u is also a dominator of v. That is, the dominator is related on path which is from the component to the system outputs.

Definition 3 (Cone [17])**.** Given a circuit, each dominator and all components dominated by it compose a cone. The dominator represents the corresponding cone.

The abstracted circuit can be obtained by abstracting the components into "top-level" cones. The abstraction process downscales the original circuit. A component may have more than one dominator and may also be a dominator of other components according to the definition of dominator. Hence, in practice, the cones used in the abstraction process may have more than one dominator. For example, c_2 is an immediate dominator of c_1, c_3 is an immediate dominator of c_2, and c_1 is not a dominator (we only consider the components c_1–c_3). Components c_1, c_2, and c_3 compose a cone and c_3 represents this cone in the abstraction process.

Definition 4 (TLD)**.** A cardinality-minimal diagnosis is TLD if it does not contain any dominated component.

Previous MaxSAT-based MBD methods (Table 1) obtain either a TLD or partially cardinality-minimal diagnoses that need consistency checking. There are two main approaches to the TLD expansion process, as described above. One is to search for subcircuits according to the dominators; then, all cardinality-minimal diagnoses can be obtained on each subcircuit to replace the corresponding dominator in TLD.

Table 1 Previous MBD to MaxSAT method

	Previous MBD to MaxSAT method (SD,Comps,Obs)
1	FindDominator()
2	CircuitAbstraction()
3	ObtainTLDbyMaxSAT()

Table 2 Previous TLD expansion process

Previous TLD expansion process (TLD)	
1	ObtainSubCircuitDiagnoses(Dominator in TLD) or ObtainDominatedComponent(Dominator in TLD)
2	IterationEnumerationReplaceTLD(SubCircuitDiagnoses) or IterationEnumerationReplaceTLD(DominatedComponent)
3	CheckConsistency(each extended TLD)

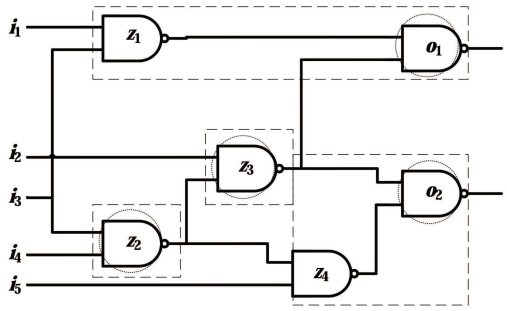

Figure 1 Example circuit, where seed-components are indicted by circles and zones are indicated by rectangles.

The other focuses on enumerating the replacements iteratively according to the domination relationships and checking consistency. Previous expansion processes on the TLD (Table 2) all need to be checked for consistency for each extended diagnosis, while our algorithm obtains all cardinality-minimal diagnoses without checking consistency in the expansion process.

3 Efficient zonal MBD with MaxSAT

In this section, we propose a new MBD method based on ZD. We first introduce the formal notions and a theorem that is used in our method. Next, we present our ZD algorithm and the propagate-extend method for obtaining all cardinality-minimal diagnoses. In addition, we describe how we obtain subset-minimal diagnoses in order of cardinality, hence obtaining the cardinality-minimal diagnoses.

3.1 Formal notions and theoretical basis

To introduce our ZD method, we first define some concepts.

Definition 5 (Seed-component)**.** Given an MBD problem $\langle \mathrm{SD}, \mathrm{Comps}, \mathrm{Obs} \rangle$, a component is a seed-component if output(c) is system output or output(c) acts as the input of at least two components.

In the example circuit shown in Figure 1, o_1 and o_2 are components directly connected to the system output and hence are both seed-components (indicated by circles). Moreover, the outputs of z_2 and z_3 both act as the inputs of two components, and therefore they are also seed-components.

Definition 6 (Zone)**.** Given an MBD problem $\langle \mathrm{SD}, \mathrm{Comps}, \mathrm{Obs} \rangle$, a disjoint partitioning $\mathrm{Comps} = Z_1 \cup Z_2 \cup \cdots \cup Z_k$ is defined, where each Z_i has only one seed-component as a root node and is called a zone.

Each zone uses the only seed-component to represent itself. Other components in the zone are expanded by searching for the nonseed-components for which the output can reach the seed-component. The process is iterated until each component belongs to a zone.

To illustrate the utility of zones, consider again the circuit shown in Figure 1, where the zones are indicated by rectangles. We first find all seed-components, each of which is then extended to a zone.

For instance, nonseed-component z_1 can reach seed-component o_1, so o_1 expands its zone from $\{o_1\}$ to $\{z_1, o_1\}$.

Definition 7 (Seed-TLD)**.** Given an MBD problem $\langle \mathrm{SD}, \mathrm{Comps}, \mathrm{Obs} \rangle$, a cardinality-minimal diagnosis is a seed-TLD if it does not contain nonseed-component.

As described above, we compute the seed-TLDs for the seed-components without considering the nonseed-components. Seed-components represents zones. Because the definition of a diagnosis is based on consistency, it would be meaningless to say that the output of a component is same as its expected output when it is abnormal. Namely, an abnormal seed-component output leads to the abnormal output of the corresponding zone. Note that previous methods [13, 14, 17, 34] focus on computing TLDs, while we use zones rather than cones to replace the whole circuit. The differences between these two methods are detailed in Subsection 3.4.

Note that we obtain seed-TLDs in order of cardinality via the MaxSAT solver and discard any diagnosis with a size that is not cardinality minimal.

Definition 8 (Zone-minimal diagnosis)**.** Given zones $Z_1, Z_2, \ldots, Z_k$, a minimal component set $\mathrm{ZDiag} \subset Z_i$ is a zone-minimal diagnosis that results in an output of Z_i that is contrary to the predicted one.

Namely, by flipping the values of the components in zone-minimal diagnosis $\mathrm{ZDiag} \subset Z_i$, we obtain the output of Z_i that is opposite to the predicted one. Let $\Delta = \{Q_1, Q_2, \ldots, Q_k\}$ be a seed-TLD that consists of k seed-components from the corresponding zones $Z_1, Z_2, \ldots, Z_k$. That is, the components in $\{Q_1, Q_2, \ldots, Q_k\}$ are abnormal, leading to abnormal outputs for $\{Z_1, Z_2, \ldots, Z_k\}$.

Definition 9 (Sensitive input)**.** Given a component c with $i_1, i_2, \ldots, i_k$ as inputs, i_m ($m \in \{1, \ldots, k\}$) is a sensitive input of c if its value can determine the output of c.

Namely, an AND gate is sensitive to 0 as an input while an OR gate is sensitive to 1 as an input. Obviously, 0 AND $R = 0$ and 1 OR $R = 1$ (where R denotes either 0 or 1).

When we search for a zone-minimal diagnosis, the number of sensitive inputs is an important factor. We explain this further in Subsection 3.3.

Theorem 1. Let $\Delta = \{Q_1, Q_2, \ldots, Q_k\}$ be a seed-TLD that consists of k seed-components from corresponding zones $Z_1, Z_2, \ldots, Z_k$. After replacing Q_i in seed-TLD with the zone-minimal diagnosis $\mathrm{ZDiag} \subset Z_i$, the resulting seed-TLD is a subset-minimal/cardinality-minimal diagnosis.

Proof. Let $S = \{Q_1, Q_2, \ldots, Q_{i-1}, \mathrm{ZDiag}, Q_{i+1}, \ldots, Q_k\}$ be a set extended by Z_i.

(S is a diagnosis.) According to Definition 8, the components in ZDiag are abnormal, leading to the abnormal output of Z_i. In addition, the components in $\{Q_1, Q_2, \ldots, Q_{i-1}, Q_{i+1}, \ldots, Q_k\}$ are abnormal, leading to the abnormal outputs of $\{Z_1, Z_2, \ldots, Z_{i-1}, Z_{i+1}, \ldots, Z_k\}$ according to Definitions 7 and 8. Namely, the components in $\{Q_1, Q_2, \ldots, Q_{i-1}, \mathrm{ZDiag}, Q_{i+1}, \ldots, Q_k\}$ are abnormal, leading to abnormal outputs of $\{Z_1, Z_2, \ldots, Z_k\}$. Hence, S is also a diagnosis.

(S is minimal.) Reduction to absurdity. Note that If ZDiag just has one component, the "minimal" proving process mentioned below means cardinality minimality. Otherwise, the "minimal" proving process mentioned below means subset minimality. Let $c \in S$. Suppose $S \setminus c$ is also a diagnosis.

(1) When $c \in \{Q_1, Q_2, \ldots, Q_{i-1}, Q_{i+1}, \ldots, Q_k\}$, the components in $\{Q_1, Q_2, \ldots, Q_{i-1}, Q_{i+1}, \ldots, Q_k\} \setminus c$ are abnormal, leading to abnormal outputs of $\{Z_1, Z_2, \ldots, Z_k\}$. In addition, the components in Q_i are abnormal, leading to abnormal outputs of Z_i. Then, $\{Q_1, Q_2, \ldots, Q_k\} \setminus c$ becomes a diagnosis. However, this contradicts Definition 7, which states that $\Delta = \{Q_1, Q_2, \ldots, Q_k\}$ is a seed-TLD.

(2) When $c \in \mathrm{ZDiag}$, the components in $\mathrm{ZDiag} \setminus c$ are abnormal, leading to the abnormal output of Z_i. However, this contradicts Definition 8, which states that ZDiag is zone-minimal diagnosis.

More importantly, our approach obtains subset-minimal diagnoses in order of cardinality. Namely, we obtain cardinality-minimal diagnoses by obtaining subset-minimal diagnoses in order of cardinality and discarding those that are not cardinality minimal.

According to Theorem 1, we can extend the seed-TLDs to all cardinality-minimal diagnoses efficiently via the propagate-extend method without checking consistency. We present its completeness later.

3.2 Zonal MBD with MaxSAT

The key idea behind our algorithm is to start by obtaining all seed-components SeedC in the circuit. We then acquire the zones of the circuit by extending the seed-components. The abstracted circuit can be obtained with zones and is used in the diagnosis process to replace the original circuit. Hence, diagnosis must be created using the components in SeedC. This technique can significantly reduce the scale of the system model, thus allowing it to compile and diagnose larger circuits.

In particular, rather than checking consistency, we propose a new method called the propagate-extend method to expand the seed-TLDs to all cardinality-minimal diagnoses, which expands the seed-TLDs directly according to Theorem 1.

We now present in detail our ZD algorithm. The pseudo code of ZD is given in Algorithm 1.

Algorithm 1 ZD: zonal diagnosis algorithm (SD, Obs)

Input: SD, Obs;
Output: D: set of cardinality-minimal diagnosis.
1: $C \leftarrow$ Components(SD);
2: Out $\leftarrow$ Outputs(Obs, SD); # find system outputs and each component outputs
3: SeedC $\leftarrow$ FindSeedComponents(C, Out);
4: Zones $\leftarrow$ GetZones(C, Out, SeedC);
5: $P \leftarrow$ Model(Zones, SD);
6: STLDs $\leftarrow$ GetSeedTLDs(P, C, SeedC);
7: $D \leftarrow \emptyset$;
8: STLD $\leftarrow$ GetOneSTLD(STLDs);
9: **while** true **do**
10: **if** all STLD $\in$ STLDs are visited **then**
11: break;
12: **else**
13: $D \leftarrow D \cup$ Extend(STLD);
14: STLD $\leftarrow$ NextSTLD(STLDs);
15: **end if**
16: **end while**

Step 1 (Find seed-components). The ZD algorithm starts by identifying all seed-components in the circuit. First, the fan-out components of every component are acquired. This is accomplished by a breadth-first traversal of the circuit starting from the outputs. Then, we identify the number of seed-components by checking the fan-out components of every component: components with two or more fan-out components are seed-components and components whose output is a system output are also seed-components. FindSeedComponents implements this procedure on line 3 of Algorithm 1.

Take Figure 1 for example. The seed-components are o_1, o_2, z_2, and z_3. Components o_1 and o_2 are connected to the system output directly. Moreover, the outputs of z_2 and z_3 both act as the inputs of two components.

Step 2 (Find zones and model the circuit). Each seed-component represents a zone. Zones are extended using seed-components by searching for the nonseed-components whose outputs can reach at least one seed-component. This procedure can be implemented by a depth-first traversal of the circuit. GetZones implements this procedure on line 4 of Algorithm 1.

After dividing the circuit into zones, the new circuit is modeled. The nonseed-components do not participate in the computation of the seed-TLDs. The CNF file is modified accordingly. The Model function implements this procedure on line 5 of Algorithm 1.

In our example in Figure 1, the circuit is divided into zones o_1, o_2, z_2, and z_3, which represent $\{o_1, z_1\}$, $\{z_2\}$, $\{z_3\}$, and $\{o_2, z_4\}$, respectively.

Step 3 (Compute seed-TLDs). After the circuit has been divided into zones, the seed-TLDs are computed for the seed-components using a MaxSAT solver. More importantly, we obtain seed-TLDs in order of cardinality and discard those diagnoses that are not cardinality minimal. Hence, the cardinality-minimal diagnoses are obtained. We added a cardinality-order constraint to the MaxSAT solver we used so that we obtain the diagnoses in order of cardinality. The seed-components are declared as soft clauses

while the rest are declared as hard clauses. Algorithm 2 implements this procedure.

In our example, given observation {1, 1, 1, 1, 1} as the input and {0, 1} as the output, $\{o_1, o_2\}$ is a seed-TLD.

Algorithm 2 GSTLDs: GetSeedTLDs(P,C,SeedC) # obtain seed-TLDs in order of cardinality and discard those diagnoses that are not cardinality-minimal

Input: P: file of CNF, C, SeedC: set of components;
Output: STLDs: set of seed-TLDs.
1: SCs ← SoftClauses(SeedC);
2: HCs ← HardClauses(C\SeedC);
3: HCs ← HCs ∪ HardClauses(P);
4: W ← HCs ∪ SCs;
5: STLDs ← ∅;
6: **while** STLD ← MaxSAT(W) **do**
7: $W \leftarrow W \cup (\infty, \neg\text{STLD})$; # avoid repetitive solutions;
8: STLDs ← STLDs ∪ STLD;
9: **end while**

Step 4 (Extend seed-TLDs). The seed-TLDs need to be extended to obtain all cardinality-minimal diagnoses. This procedure is implemented in Algorithm 3. Previous studies on extending TLDs [13, 14, 17, 34] focus on iterative enumeration of the replacements and need to check consistency, which is a time-consuming process. We propose a new method for extending the seed-TLDs called propagate-extend. This method expands the seed-TLDs directly without having to check consistency. Its details are given in Subsection 3.3.

Algorithm 3 Extend(STLD)

Input: STLD: set of components;
Output: ETLDs: set of extended seed-TLDs.
1: Comp ← GetOneComp(STLD); # the Comp is a seed-component
2: **while** true **do**
3: **if** all Comp ∈ STLD are visited **then**
4: break;
5: **else**
6: SeedComp ← Comp;
7: ETLD ← PropagateExtend(SeedComp,Comp); # the first Comp is to retain the seed-component in each iteration of PropagateExtend
8: ETLDs ← ETLDs ∪ ETLD;
9: Comp ← NextComp(STLD);
10: **end if**
11: **end while**

3.3 Propagate-extend method

The propagate-extend method extends the seed-TLDs by traversing the zones depth-first according to Theorem 1 rather than calling a solver to check the consistency. The seed-component is its input. This method is outlined in Algorithm 4 and described as follows.

For the current component Comp under consideration, the PropagateExtend algorithm starts by checking whether the component has fan-in components. If this is the case, the algorithm calls the FlipValue function for each of its fan-in components.

The algorithm distinguishes two different cases.

(1) Comp has no sensitive inputs. In this case, for each fan-in component, the algorithm determines whether it can form a zone-minimal diagnosis that can replace the current component (lines 2–9). In addition, this propagate-extend method is executed iteratively (line 6). If Comp is a fault with no sensitive inputs, flipping each non-sensitive input changes the output of Comp to the opposite of its expected value. In addition, we check whether this flip operation causes the output of the seed-component to change from its expected value.

Algorithm 4 PropagateExtend: diagnosis of zone (SeedComp,Comp)

Input: SeedComp: a seed-component, Comp: a component;
Output: ZDiag: set of cardinality-minimal/subset-minimal diagnosis.

```
 1: if HasFaninComp(Comp) = true then
 2:   if Comp has no sensitive inputs then
 3:     FIComp ← GetOneNonSensitiveFIComp(Comp);
 4:     if flipping the value of FIComp can make the output of SeedComp opposite with its expected value then
 5:       ZDiag ← ZDiag ∪ FIComp;
 6:       FIZDiag ← PropagateExtend(SeedComp,FIComp);
 7:       ZDiag ← ZDiag ∪ FIZDiag;
 8:       FIComp ← NextSensitiveFIComp(Comp);
 9:     end if # the later steps (lines 10–26) will be passed when we want cardinality-minimal diagnoses
10:   else
11:     FIComps ← GetAllSensitiveFIComp(Comp);
12:     if flipping all value of FIComps can make the output of SeedComp opposite with its expected value then
13:       ZDiag ← ZDiag ∪ {FIComps};
14:     end if
15:     SubFIComp ← GetOneComp(FIComps);
16:     SubZDiag ← ∅;
17:     while true do
18:       if all SubFIComp ∈ FIComp are visited then
19:         break;
20:       else
21:         SubZDiag ← SubZDiag ∪ Propagate(SubFIComp); # search for the minimal sets that flipping the value of
            them in it can lead to the output of SubFIComp opposite with its expected value
22:       end if
23:     end while
24:     if flipping all value of each set CartesianProduct(SubZDiag) can make the output of SeedComp opposite with its
        expected value then
25:       ZDiag ← ZDiag ∪ CartesianProduct(SubZDiag);
26:     end if
27:   end if
28: end if
```

(2) Comp has sensitive inputs. In this case, the algorithm determines whether the sensitive fan-in components as a whole can form a zone-minimal diagnosis that can replace the current component (lines 10–26). If Comp is a fault with sensitive inputs, only flipping all sensitive fan-in component will change the output of Comp from its expected value. We also check whether this flip operation can change the output of the seed-component from its expected value. In addition, for each sensitive fan-in component, we search for the minimal set such that flipping the values of the components in the set changes the output of the fan-in component from the expected value on line 21. This step is similar to the propagate-extend process, but it only propagates without extending. We need to check whether each set in the Cartesian product of these minimal sets can become a zone-minimal diagnosis.

In addition, the second case means that the current diagnosis is not cardinality minimal. Hence, we discard this extended diagnosis. Note that these steps (lines 10–26) are executed when we want cardinality-minimal diagnoses.

To illustrate the process of this algorithm, consider the partial circuit given in Figure 2, which denotes a zone. Obviously, z_9, an AND gate, is the only seed-component of the zone. In this zonal structure, we regard the seed-component as the root node of a tree structure, and z_6, z_7, and z_8 are the fan-in nodes of z_9.

The algorithm finds the number of sensitive fan-ins to z_9 because it is an AND gate. The rest of the algorithm proceeds according to two cases. For our example, we introduce the two cases separately.

(1) Suppose the values of f_{17}, f_{18}, and f_{19} are 1, 1, and 1, respectively; thus, z_9 has no sensitive inputs.

The algorithm then flips the value f_{17} of z_6. If this flip makes output f_{20} of seed-component z_9 change from its expected value, then f_{17} is flipped and z_6 comprises the zone-minimal diagnosis according to Theorem 1. The algorithm executes iteratively using z_6 and seed-component z_9 as its inputs. Note that the following iteration may be the first case or the second case, depending on the number of sensitive

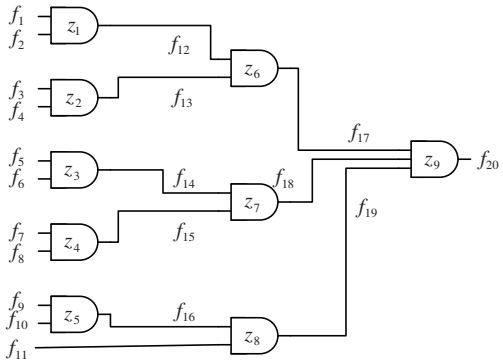

Figure 2 Example of a zone in a circuit.

inputs of z_6.

The algorithm then flips the value of f_{18}. If this makes f_{20} of seed-component z_9 change from its expected value, then f_{18} is flipped, and z_7 composes the zone-minimal diagnosis according to Theorem 1. The algorithm then executes iteratively with z_7 and seed-component z_9 as inputs. As above, the next iteration may be the first or second case, depending on the number of sensitive inputs of z_7.

The algorithm then flips the value of f_{19}. If this changes f_{20} of seed-component z_9 from its expected value, then f_{19} is flipped, and z_8 composes the zone-minimal diagnosis according to Theorem 1. The algorithm executes iteratively using z_8 and seed-component z_9 as inputs. As above, the next iteration may be the first or second case, depending on the number of sensitive inputs of z_8.

(2) Suppose the values of f_{17}, f_{18}, and f_{19} are 0, 0, and 1, respectively; thus, z_9 has two sensitive inputs f_{17} and f_{18}.

Then, the algorithm tries to flip the value f_{17} and f_{18}. If this flipping makes the output f_{20} of seed-component z_9 change from its expected value, z_6 and z_7 compose the zone-minimal diagnosis according to Theorem 1. We then search for the minimal sets for z_6 such that flipping all the values of the members of the set changes the output of z_6 from its expected value. In addition, z_7 undergoes same process. We check whether each set in the Cartesian product of these minimal sets could be a zone-minimal diagnosis on line 24 of Algorithm 4.

Note that when zone-minimal diagnosis consisting of a single component replaces a seed-component, the extended diagnosis remains cardinality minimal. When zone-minimal diagnosis consisting of multiple components replaces a seed-component, this case will be discarded if we only want cardinality-minimal diagnoses.

Completeness of the propagate-extend method. The propagate-extend method searches for a zone-minimal diagnosis to replace the seed-component in each zone. Note that the seed-TLD is cardinality minimal, as above. Finding a zone-minimal diagnosis starts by verifying the number of sensitive inputs. When a zone-minimal diagnosis has only one component, the extended diagnosis is also a cardinality-minimal diagnosis according to Theorem 1. We traverse the zone depth-first to ensure we access the components in order and explore all possibilities. Hence, we can obtain all cardinality-minimal diagnoses. In contrast, when the zone-minimal diagnosis has more than one component, the Cartesian product operation in the propagate-extend method ensures that the power set of the components is enumerated. In this case, we obtain a diagnosis that is subset minimal rather than cardinality minimal. In addition, this process is executed when we want cardinality-minimal diagnoses only.

3.4 Differences with dominator and TLD

To reduce the scale of the circuit, our method uses an approach that is different from previous methods that are based on the domination relationship. We extend the seed-components to zones, and the whole circuit is eventually replaced by zones. Some specific differences are as follows.

D1. Seed-components vs. dominators. From our definition, a seed-component is related to the number of fan-out components. However, the concept of a dominator [17] is related to the paths from

a component to the system output. The intersections of these paths are dominators. Obviously, the concept of a dominator is stricter.

D2. Seed-TLDs vs. TLDs. Our method uses the seed-TLD, while previous methods use the TLD.

D3. Zones vs. cones and sections. The concept of a cone is based on a dominator while our zone is based on a seed-component. Some components in a cone that can reach the same circuit system output compose a section [13, 14]. In contrast, our zone is based on a seed-component, which is related to the number of fan-out components. Namely, a zone only has one output, which is the output of corresponding seed-component. Moreover, zones and cones are used to downscale the circuit while a section is used to estimate the size of the cardinality-minimal diagnosis.

More importantly, the propagate-extend method cannot be used on cones. The dominator is the component that appears in each path from the component to system outputs. It is possible that some components in a cone cannot reach the dominator via the internal cone connections. Namely, a cone does not necessarily have complete internal connections. However, our propagate-extend method is based on the complete internal connections within a zone. Hence, the propagate-extend method cannot be used on cones because this case could occur.

We consider a simple example. Suppose component c_1 has three paths to the system output $\{c_1, c_2, c_3, c_4, c_7\}$, $\{c_1, c_5, c_3, c_4, c_7\}$, and $\{c_1, c_6, c_3, c_4, c_7\}$. Further, the output of c_1 only reaches c_2, c_5, and c_6 directly, the output of c_3 only reaches c_4 directly, the output of c_4 only reaches c_7 directly, and the output of c_7 reaches more than one component directly. The outputs of c_2, c_5, and c_6 only reach c_3 directly. In addition, we only consider the components c_1–c_7.

In the cone method, obviously, c_3 is the immediate dominator of c_1 because all paths from c_1 to the system outputs contain c_3, c_4, and c_7 and both c_4 and c_7 are a dominator of c_3. Moreover, c_4 is the immediate dominator of c_3 and c_7 is the immediate dominator of c_4. Hence, c_1, c_3, c_4, and c_7 compose a "top-level" cone in the abstraction process and c_7 represents this cone. In this cone, c_3, c_4, and c_7 have complete connections while c_1 and c_3 do not. Namely, the internal connections in the cone are not complete.

In contrast, in our approach, c_1 and c_7 are seed-components. Namely, c_1 is in a different zone from c_3, c_4 and c_7. In addition, c_2, c_5, c_6, c_3, c_4, and c_7 compose a zone. The internal connections in this zone are complete. Our propagate-extend approach is based on such complete internal connections. Hence the propagate-extend approach cannot be used on cones.

D4. Obtaining all cardinality-minimal diagnoses. Previous methods for obtaining all cardinality-minimal diagnoses focus on extending TLDs, which are subcircuit diagnosis, and iterative enumeration of replacements. A drawback of these techniques is the time consumed to check consistency. Our method extends the seed-TLDs without checking consistency. It only takes 0.1 s to extend 100 cardinality-minimal diagnoses.

4 Experimental evaluations

This section evaluates our ZD approach to computing all cardinality-minimal diagnoses. We first compare the scalability of our approach with that of hierarchical diagnosis [17], which is a state-of-the-art approach for downscaling circuits in the diagnosis problem. We then compare our approach to a state-of-the-art MBD solver for computing all cardinality-minimal diagnoses as well as a degraded version of our approach that works without the strategies presented in this paper.

4.1 Experiment preliminaries

Implementation. Our approach was implemented in C++ and compiled with the g++ compiler. Moreover, we adopted the open-wbo-inc (wboinc) solver [35] as the MaxSAT solver in our approach, as it is one of the best performing MaxSAT solvers for partial MaxSAT instances. Note that we use the MaxSAT solver to obtain subset-minimal diagnoses in order of cardinality to obtain cardinality-minimal

Table 3 Comparing reducing power of cones and zones on ISCAS-85 circuits

Circuit	Components	Hierarchical diagnosis		ZD	
		Health vars	Reduce (%)	Zones	Reduce (%)
c432	160	59	63.1	65	59.4
c499	202	58	71.3	66	67.3
c880	383	77	79.9	132	65.5
c1355	546	58	70.3	266	51.3
c1908	880	160	57.5	392	55.5
c2670	1193	167	86.0	399	66.6

diagnoses. Namely, we obtain cardinality-minimal diagnoses by obtaining subset-minimal diagnoses in order of cardinality and discarding those that are not cardinality minimal.

Competitors. We first compare the scalability of our approach with that of hierarchical diagnosis [17], which represents the best method for scale reducing. We then focus on comparing our solver with the latest version of SATbD [13,14], which was recently shown to outperform most complete MBD approaches using SAT.

Note that we do not compare our approach with MaxSAT-based MBD algorithms because previous studies on MBD using MaxSAT focus on obtaining one or partial cardinality-minimal diagnosis, while our approach acquires all cardinality-minimal diagnoses. Instead, a simple complete algorithm for solving MBD with the MaxSAT solver open-wbo-inc is also implemented, called MM, which does not contain any strategies.

Benchmarks. We ran 5506 instances on a series of ISCAS-85 circuits using random observations. For each circuit, we randomly generated the expected observations according to the normal behavior of the circuit, then we flipped t outputs randomly, for values of t from 1 to 8, so as to obtain abnormal observations. Note that, in most cases, neither our approach nor SATbD compute all cardinality-minimal diagnoses within the time limit for the larger circuits (i.e., c5315, c6288, and c7552, where we used a very large diagnosis number limit, such as 10^5 or 10^6). Hence, we do not list these results.

Experimental setup. The experiments were performed on a 32-bit Ubuntu machine equipped with one Intel i5-4590 3.30 GHz processor and 1 GByte of physical memory. For all experiments, the time limit was set to 1500 s.

4.2 Experimental results

This subsection reports the results of our experiments, which consist of two parts: scalability and efficiently.

(a) Scalability. First, we compare the scalability of our approach with that of [17] on ISCAS85 circuits. As the method in [17] only returns one TLD rather than all cardinality-minimal diagnoses, we cannot compare our method with it directly. Instead, we compare the reducing power of our abstracting method based on zones with the method based on "health vars" in [17].

The experimental results are reported in Table 3. Because [17] reports the scalability for six circuits, we compare the reducing power for these six circuits. The results show that the reducing power of our abstracting approach is weaker than the dominator-based approach. However, our propagate-extend method in this paper is particularly suitable for the zone diagnosis approach. In addition, it cannot be used on cones.

(b) Efficiency. A recent study presented empirical evidence suggesting that a "direct" search for diagnoses is often better than conflict-directed diagnosis algorithms [36]. Hence, we compare the efficiency of our approach with SATbD [13, 14] on ISCAS85 circuits. SATbD is based on a complete algorithm for finding all cardinality-minimal diagnoses directly. It was recently shown to outperform most complete MBD approaches using SAT.

We tested 5506 instances on ISCAS85 circuits. The experimental results (Table 4) indicate that the efficiency of our approach outperforms SATbD for most instances on the ISCAS85 circuits in Table 3 when the diagnosis number is limited to 10^1, 10^2, 10^3, and 10^4. As can be observed, the runtime of

Table 4 Run time comparison for the ISCAS-85 circuits [a)]

Circuit	Diagnosis number ⩽ 10			Diagnosis number ⩽ 100		
	SATbD	MM	ZD	SATbD	MM	ZD
c432	0	0.04	**0**	0.18	0.21	**0.06**
c499	0.01	0.05	**0**	0.16	0.36	**0**
c880	0.02	0.02	**0**	**0.08**	0.49	0.09
c1355	0.05	0.11	**0**	0.09	0.96	**0.08**
c1908	0.34	0.26	**0.02**	0.64	1.76	**0.41**
c2670	0.24	0.47	**0.07**	0.74	4.46	**0.71**
	Diagnosis number ⩽ 1000			Diagnosis number ⩽ 10000		
	SATbD	MM	ZD	SATbD	MM	ZD
c432	**0.24**	0.54	0.27	0.42	1.41	**0.31**
c499	0.74	1.01	**0.41**	1.74	1.84	**0.95**
c880	0.42	1.41	**0.27**	1.46	2.97	**0.93**
c1355	0.57	1.52	**0.45**	1.82	6.85	**1.02**
c1908	**0.84**	3.11	0.90	3.33	12.75	**2.98**
c2670	1.32	10.87	**1.09**	7.65	36.49	**4.96**

a) Bold: the winner for each scenario.

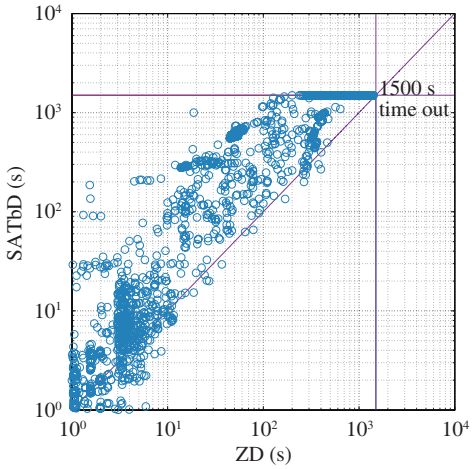

Figure 3 (Color online) Run time comparison for the ISCAS-85 circuits (ZD vs. SATbD).

our approach (Table 4) is always less than 1.0 s. Hence, we tested ISCAS85 circuits without limiting the number of diagnoses. The experimental results are reported in Figure 3. As can be observed, the ZD approach is mostly one to two orders of magnitude faster than SATbD. In addition, from the data in Figure 3, we randomly selected one case for each circuit to compare the runtime in order of circuit size (Figure 4). As the number of diagnosis increases with the circuit scale, the impact of the expansion process becomes more significant. Namely, the performance improvements of the ZD method are more substantial as the circuits become larger. Moreover, the runtimes for some circuits are over the time limit when SATbD is used, while our approach can deal with these circuits.

Typically, the diagnosis problem becomes harder as the number of diagnoses increases. The plot in Figure 5 shows the results for c3540 for different numbers of diagnoses. As can be observed, our ZD approach outperforms SATbD and can return larger numbers of diagnoses for a particular problem.

5 Conclusion

This paper presented a novel approach to solving MBDs with MaxSAT on a standard ISCAS-85 benchmark. This method downscales the problem and finds all cardinality-minimal diagnoses efficiently. We use open-wbo-inc, a state-of-the-art MaxSAT solver. First, the concept of zones is used to downscale the

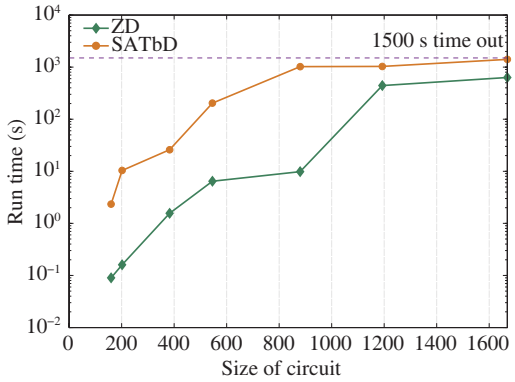

Figure 4 (Color online) Run time comparison on ISCAS85 circuits.

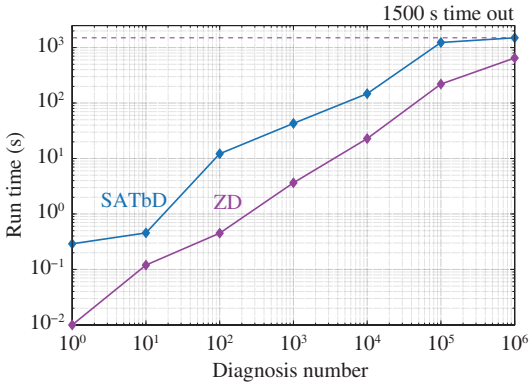

Figure 5 (Color online) Run time comparison on c3540.

circuit. Second, the propagate-extend method is used to extend the seed-TLDs directly. The experimental evaluation focused on the ISCAS-85 benchmarks. We compared the efficiency of our approach with that of SATbD and the scalability of our approach with that of hierarchical diagnosis. The results are unequivocal: our approach outperforms SATbD in terms of run time, one to two orders of magnitude within time limit.

In the future, we would like to seek further optimizations using the ZD approach.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. 61672261, 61502199, 61402196, 61272208).

References

1 Console L, Dressler O. Model-based diagnosis in the real world: lessons learned and challenges remaining. In: Proceedings of the 16th International Joint Conference on Artificial Intelligence, Stockholm, 1999. 1393–1400
2 Xu K, Li W. Many hard examples in exact phase transitions. Theory Comput Sci, 2006, 355: 291–302
3 Wang Y Y, Ouyang D T, Zhang L M, et al. A novel local search for unicost set covering problem using hyperedge configuration checking and weight diversity. Sci China Inf Sci, 2017, 60: 062103
4 Gao J, Wang J N, Yin M H. Experimental analyses on phase transitions in compiling satisfiability problems. Sci China Inf Sci, 2015, 58: 032104
5 Geng X N, Ouyang D T, Zhang Y G. Model-based diagnosis of incomplete discrete-event system with rough set theory. Sci China Inf Sci, 2017, 60: 012205
6 Wang X Y, Jiang S J, Gao P F, et al. Cost-effective testing based fault localization with distance based test-suite reduction. Sci China Inf Sci, 2017, 60: 092112
7 Zhou J P, Yin M H, Li X T, et al. Phase transitions of expspace-complete problems: a further step. Int J Found Comput Sci, 2012, 23: 173–184
8 de Kleer J, Mackworth A K, Reiter R. Characterizing diagnoses and systems. Artif Intel, 1992, 56: 197–222
9 De K J. US Patent, 9563525, 2017-02-07
10 Jannach D, Schmitz T, Shchekotykhin K. Parallelized hitting set computation for model-based diagnosis. In: Proceedings of the 29th AAAI Conference on Artificial Intelligence, Austin, 2015. 1503–1510
11 Williams B C, Ragno R J. Conflict-directed A* and its role in model-based embedded systems. Discrete Appl Math, 2007, 155: 1562–1595
12 Feldman A, Provan G, de Kleer J, et al. Solving model-based diagnosis problems with max-sat solvers and vice versa. In: Proceedings of the 21th International Workshop on Principles of Diagnosis, Portland, 2010. 185–192
13 Metodi A, Stern R, Kalech M, et al. Compiling model-based diagnosis to boolean satisfaction. In: Proceedings of the 26th AAAI Conference on Artificial Intelligence, Toronto, 2012. 793–799
14 Metodi A, Stern R, Kalech M, et al. A novel sat-based approach to model based diagnosis. J Artif Intell Res, 2014, 51: 377–411
15 Zhang J, Ma F F, Zhang Z Q. Faulty interaction identification via constraint solving and optimization. In: Proceedings of the 15th Theory and Applications of Satisfiability Testing, Trento, 2012. 186–199
16 Darwiche A. Decomposable negation normal form. J ACM, 2001, 48: 608–647
17 Siddiqi S, Huang J B. Hierarchical diagnosis of multiple faults. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, 2007. 581–586
18 Stern R, Kalech M, Feldman A, et al. Exploring the duality in conflict-directed model-based diagnosis. In: Proceedings

of the 26th AAAI Conference on Artificial Intelligence, Toronto, 2012. 828–834
19 Smith A, Veneris A, Ali M F, et al. Fault diagnosis and logic debugging using Boolean satisfiability. IEEE Trans Comput-Aided Design, 2005, 24: 1606–1621
20 Mangassarian H, Veneris A, Safarpour S, et al. A performance-driven QBF-based iterative logic array representation with applications to verification, debug and test. In: Proceedings of the 26th International Conference on Computer-Aided Design, San Jose, 2007. 240–245
21 Suelflow A, Fey G, Bloem R, et al. Using unsatisfiable cores to debug multiple design errors. In: Proceedings of the 18th ACM Great Lakes Symposium on VLSI, Orlando, 2008. 77–82
22 Safarpour S, Veneris A. Abstraction and refinement techniques in automated design debugging. In: Proceedings of the 10th Design, Automation and Test in Europe Conference and Exhibition, Nice, 2007. 1182–1187
23 Gao J, Li R Z, Yin M H. A randomized diversification strategy for solving satisfiability problem with long clauses. Sci China Inf Sci, 2017, 60: 092109
24 Liu J, Xu K. A novel weighting scheme for random k-SAT. Sci China Inf Sci, 2016, 59: 092101
25 Safarpour S, Mangassarian H, Veneris A, et al. Improved design debugging using maximum satisfiability. In: Proceedings of the 7th Formal Methods in Computer Aided Design, Austin, 2007. 13–19
26 Kutsuna T, Sato S, Chujo N. Diagnosing automotive control systems using abstract model-based diagnosis. In: Proceedings of the 20th International Workshop on Principles of Diagnosis, Stockholm, 2009. 99–105
27 Chen Y B, Safarpour S, Veneris A, et al. Spatial and temporal design debug using partial MaxSAT. In: Proceedings of the 19th ACM Great Lakes Symposium on VLSI, Boston Area, 2009. 345–350
28 Cai S W, Luo C, Thornton J, et al. Tailoring local search for partial MaxSAT. In: Proceedings of the 28th AAAI Conference on Artificial Intelligence, Quebec, 2014. 2623–2629
29 Marques-Silva J, Janota M, Ignatiev A, et al. Efficient model based diagnosis with maximum satisfiability. In: Proceedings of the 24th International Joint Conference on Artificial Intelligence, Buenos Aires, 2015. 1966–1972
30 Kirkland T, Mercer M. A topological search algorithm for ATPG. In: Proceedings of the 24th ACM/IEEE Design Automation Conference, Miami Beach, 1987. 502–508
31 Huang J B, Darwiche A. On compiling system models for faster and more scalable diagnosis. In: Proceedings of the 20th National Conference on Artificial Intelligence, London, 2005. 300–306
32 Reiter R. A theory of diagnosis from first principles. Artif Intel, 1987, 32: 57–95
33 de Kleer J, Williams B C. Diagnosing multiple faults. Artif Intel, 1987, 32: 97–130
34 Siddiqi S. Computing minimum-cardinality diagnoses by model relaxation. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence, Barcelona, 2011. 1087–1092
35 Martins R, Joshi S, Manquinho V, et al. Incremental cardinality constraints for MaxSAT. In: Proceedings of the 20th International Conference on Principles and Practice of Constraint Programming, Lyon, 2014. 531–548
36 Nica I, Pill I, Quaritsch T, et al. The route to success-a performance comparison of diagnosis algorithms. In: Proceedings of the 22th International Joint Conference on Artificial Intelligence, Beijing, 2013. 1039–1045