Computing Minimum-Cardinality Diagnoses by Model Relaxation

Sajjad Siddiqi
National University of Sciences and Technology (NUST)
Islamabad, Pakistan
sajjad.ahmed@seecs.edu.pk

Abstract

We propose a new approach based on model relaxation to compute *minimum-cardinality diagnoses* of a (faulty) system: We obtain a relaxed model of the system by splitting nodes in the system and compile the *abstraction* of the relaxed model into DNNF. Abstraction is obtained by treating self-contained sub-systems called *cones* as single components. We then use a novel branch-and-bound search algorithm and compute the abstract minimum-cardinality diagnoses of the system, which are later refined hierarchically, in a careful manner, to get all minimum-cardinality diagnoses of the system. Experiments on ISCAS-85 benchmark circuits show that the new approach is faster than the previous state-of-the-art hierarchical approach, and scales to all circuits in the suite for the first time.

1 Introduction

Given an (abnormal) *observation* of a system, a *diagnosis* of the system is a set of system components that if assumed as broken and the rest as healthy explains the abnormal behavior. A *minimum-cardinality diagnosis* is a diagnosis of the system that declares the least number of components to be broken.

In *model-based diagnosis* [Reiter, 1987] a model of the system is maintained against which the observed (abnormal) behavior of the system can be checked to infer diagnoses. The previous technique to compute minimum-cardinality diagnoses maintains an abstract model of the system, and can efficiently compute diagnoses in a hierarchical fashion [Siddiqi and Huang, 2007]. However, the technique relies on successful compilation of the system abstraction into DNNF [Darwiche, 2001], which can be prohibitive for large systems.

We propose a novel approach, based upon *model relaxation*, which scales diagnosis to very large systems. If the abstraction of a system cannot be compiled, we obtain a relaxed model of the system, through *node-splitting* [Choi *et al.*, 2007], and compile the abstraction of the relaxed model instead. We then perform a two-stage branch-and-bound search to compute the abstract minimum-cardinality diagnoses of the relaxed model. The computed diagnoses form a super-set of the minimum-cardinality diagnoses of the abstraction of the original system, which are later refined hierarchically to get all minimum-cardinality diagnoses of the system.

The two stages of the search are as follows: First, we perform branch-and-bound search in the space of assignments to *split variables* and compute the minimum cardinality of diagnoses. At each search node, the DNNF compilation of the relaxed model is evaluated under the current partial assignment to split variables. At leaf nodes we get candidate minimum cardinalities and elsewhere bounds to prune the search. In the second stage, we consider two kinds of branch-and-bound search strategies, one that involves search in the space of assignments to split variables, and the other involves search in the space of assignments to *health variables* (i.e., variables that model the health of system components). We then, systematically, combine the two strategies by searching in both spaces simultaneously to gain the strengths of both strategies.

We deal with intricacies in computing the diagnoses correctly, and propose novel techniques to improve the efficiency of search algorithms, including the novel variable and value ordering heuristics and techniques to avoid redundancy.

2 Background

In *model-based diagnosis* [Reiter, 1987] a system to be diagnosed consisting of a set of components $\mathbf{C}$ is modeled as first order formulas Δ representing the behavior and the connections between various components of the system. Let okX be a propositional variable representing the health of a component $X \in \mathbf{C}$, we can write sentences of the form $okX \rightarrow$NormalBehavior(X) for all $X \in \mathbf{C}$ to model the normal behavior of the system.

Given a set of *observations* β (a set of propositions), a *consistency-based diagnosis* of $(\Delta, \mathbf{C}, \beta)$ is defined to be a set $\mathbf{D} \subseteq \mathbf{C}$ such that $\Delta \cup \beta \cup \{okC | C \in \mathbf{C} \backslash \mathbf{D}\} \cup \{\neg okC | C \in \mathbf{D}\}$ is consistent. The cardinality of a diagnosis $\mathbf{D}$ is the number of broken components in $\mathbf{D}$ or equivalently $|\mathbf{D}|$. A diagnosis $\mathbf{D}$ of $(\Delta, \mathbf{C}, \beta)$ is of minimum cardinality if there exists no other diagnosis $\mathbf{E}$ of $(\Delta, \mathbf{C}, \beta)$ such that $|\mathbf{E}| < |\mathbf{D}|$.

We assume a particular fault model for every component, in which a component always outputs a wrong value if it fails. These can be described by sentences of the form $\neg okX \rightarrow \neg$NormalBehavior$(X)$ for each component X. Such fault models allow stronger inference on the system model, while the minimum-cardinality diagnoses with or without fault models remain the same in this case. We also

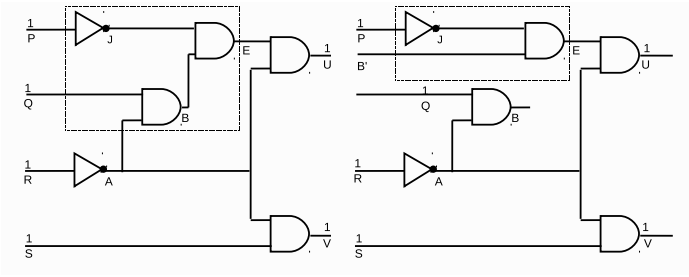

Figure 1: A circuit with abnormal observation $P \wedge Q \wedge R \wedge S \wedge U \wedge V$ (left). A split circuit obtained from this circuit after splitting B according to E (right). The binary gates are AND gates and unary gates are NOT gates.

assume that inputs and outputs of the system are observed and the internal variables are non-observable. For example, the NOT gate J in the circuit in Figure 1 (left) can be modeled as $okJ \rightarrow (P \leftrightarrow \neg J), \neg okJ \rightarrow (P \leftrightarrow J)$. Given the observation $P \wedge Q \wedge R \wedge S \wedge U \wedge V$, the minimum cardinality of diagnoses of this circuit is 2 and one minimum-cardinality diagnosis is $\{J, A\}$.

System models can be compiled into a graph-based representation of propositional theories known as *decomposable negation normal form* (DNNF), which exploits system structure to compactly represent the functionality of the system, and supports efficient computation of common diagnostic queries. Once a system is compiled into DNNF all minimum-cardinality diagnoses can be computed in time polynomial in the size of the DNNF [Darwiche, 2001].

If a system is too large to compile, one can obtain a relaxed, easy to compile, model of the system by splitting nodes in the system (from graph point of view) [Pipatsrisawat and Darwiche, 2007; Choi *et al.*, 2007]. Let N be the system graph where each component and every primary input of the system are represented with nodes. A node X is a parent of a node Y, and Y is a child of X, if the output of Y is an input of X in the system. *Node splitting* creates a *clone* $\hat{X}$ of a node X such that $\hat{X}$ inherits some of the parents of X. Formally:

Definition 1 (Node Splitting). *Let X be a node in a system graph N with parents $\mathbf{Y}$. We say that X is* **split according to parents $\mathbf{Z} \subseteq \mathbf{Y}$** *when it results in a graph N' that is obtained from N as follows: The edges outgoing from X to its parents $\mathbf{Z}$ are removed, and a new primary input node $\hat{X}$ (having no children) is added to the system graph with nodes $\mathbf{Z}$ as its parents [Choi* et al.*, 2007].*

When X is split according to its every parent it is said to be *fully split*, in which case X becomes a primary output of the new system. For example, in Figure 1 (right) B has been split according to its only parent E, and $\hat{B}$ is a clone of B. B becomes a primary output of the new circuit.

We say that if $\mathbf{e}$ is an assignment to variables in system represented by N, then $\overrightarrow{\mathbf{e}}$ is the compatible assignment to corresponding clones in system represented by N', i.e a variable and its clones (if any) are assigned the same value. For example, if $\mathbf{e} = \{B = b\}$, then $\overrightarrow{\mathbf{e}} = \{\hat{B} = b\}$.

3 Previous Work

In previous work, the abstraction-based hierarchical technique to compute minimum-cardinality diagnoses [Siddiqi and Huang, 2007] aims at reducing the required number of health variables in the model, and scales the baseline compilation-based diagnosis approach. It exploits self-contained sub-systems referred to as *cones* that can be treated as single components. One needs only a single health variable to model the health of all components in a cone. For example, in the circuit in Figure 1 (left), E is a cone with inputs $\{P, Q, A\}$ and output E. A cone may contain further cones and so on the phenomenon leads us to a hierarchy of cones. When all maximal cones in a system $\mathbf{C}$ are treated as single components we obtain an abstraction $\mathbf{A_C}$ of the system, where a maximal cone is one that is either contained in no other cone or contained in exactly one other cone which is the whole system. Formally:

Definition 1 (Abstraction of System). *Given a system $\mathbf{C}$, let $\bar{\mathbf{C}} = \mathbf{C}$ if $\mathbf{C}$ has a single output; otherwise let $\bar{\mathbf{C}}$ be $\mathbf{C}$ augmented with a dummy component collecting all outputs of $\mathbf{C}$. Let O be the only output of $\mathbf{C}'$. The abstraction $\mathbf{A_C}$ of system $\mathbf{C}$ is then the set of components $X \in \mathbf{C}$ such that X is not dominated in $\bar{\mathbf{C}}$ by any component other than X and O [Siddiqi and Huang, 2007].*

For example, the abstraction of the circuit given in Figure 1 (left) contains the gates $\{U, V, E, A\}$, excluding the set of gates $\{J, B\}$ which become part of the cone E. The size of an abstraction is the number of components contained in it.

The set of minimum-cardinality diagnoses computed from the compilation of the abstraction is always the sub-set of the minimum-cardinality diagnoses of the system. If none of the diagnoses mention the possibility of a cone to be faulty, then the computed set of diagnoses is complete. Otherwise, we note that a cone mentioned in an abstract diagnosis (i.e., the diagnosis of the abstraction) represents a set of possible single faults in the cone (the minimum cardinality of diagnoses of a cone is always 1). These single faults, once correctly diagnosed, can be substituted for the cone, one by one, in the abstract diagnosis to generate a set of global minimum-cardinality diagnoses that are guaranteed to be valid. Doing the same for every cone in every abstract diagnosis and those generated from them gives us the complete set of minimum-cardinality diagnoses.

Suppose that a cone X is declared to be faulty in an abstract diagnosis $\mathbf{D}$. X can be diagnosed hierarchically by again diagnosing the abstraction of X, once a correct abnormal observation at the inputs and output of X has been computed. For this purpose, first the values of system inputs (given in the observation) are propagated into the system assuming every component is healthy. Since, one has to diagnose X under the assumption that all of the components $G_i \in \mathbf{D}$ are faulty, the fault effect of all G_is has to be propagated into the system as well. Therefore, the set of components in $\mathbf{D}$ are placed in decreasing order of their depths in the system, as deeper faults must be propagated before the less deeper ones. The fault effect of a component is propagated by changing its output to the abnormal value and propagating its effect. Doing this for all $G_i \in \mathbf{D}$ separately in the order in which they appear in $\mathbf{D}$

sets the required observation for the cone X.

For example, one minimum-cardinality diagnosis of the circuit in Figure 1 (left) is $\mathbf{D} = \{E, A\}$, which declares the cone E as faulty. To compute an abnormal observation at the inputs and output of E (a valuation of variables $\{P, Q, A, E\}$), we re-order gates in $\mathbf{D}$ as $\{A, E\}$ (gate A is deeper than E). We then propagate the circuit inputs into the model and get the valuation $P \wedge Q \wedge \neg A \wedge \neg E$. We then propagate the faults declared in $\mathbf{D}$ one by one after flipping the values of A and then E and get the required valuation $\alpha = P \wedge Q \wedge A \wedge E$. The cone E is then separately diagnosed under the observation α. One diagnosis of E is given by $\{J\}$. Substituting J for E in $\mathbf{D}$ we get $\{A, J\}$, which is another minimum-cardinality diagnosis of the circuit.

A drawback of this approach is that compilation can still become a bottleneck if a system model remains very large even after abstraction has been applied on it. Below we describe the new approach we have proposed to circumvent this limitation in scalability.

4 Diagnosis by Model Relaxation

If the abstraction of a system cannot be compiled, we obtain a relaxed model of the system through node-splitting and diagnose the system using the relaxed model: We keep splitting nodes in a system until its *treewidth* falls to a preset target at which point it can be easily compiled. Once the abstraction of the split system has been compiled minimum-cardinality diagnoses of the abstraction can be computed using a two stage branch-and-bound search. The complete set of the minimum-cardinality diagnoses can be obtained after carefully diagnosing cones declared faulty in abstract diagnoses.

However, node-splitting can change the abstraction of a system. It may expose components that were previously contained in cones. For example, the abstraction of the split circuit in Figure 1 (right) now contains one more gate namely B (the original abstraction contained gates $\{U, V, A, E\}$). The new circuit has one more primary output B and one more primary input $\hat{B}$. The cone E has also changed as it has lost the gate B and its inputs are $\{P, \hat{B}\}$. Hence the set of components in the abstraction of the split system is a super-set of the set of components in the abstraction of the original system. Intuitively, the set of abstract minimum-cardinality diagnoses of the the split system if correctly computed would be the super-set of that of the original system.

We now describe the new diagnosis algorithm and also discuss additional techniques that we have introduced to correctly compute the required set of diagnoses.

4.1 Search for Minimum Cardinality

An interesting property of a split system is that under the given observation the minimum cardinality with respect to the split system gives a lower bound on the minimum cardinality with respect to the original system [Pipatsrisawat and Darwiche, 2007]. Hence, if Δ and Δ' are models of the original and the split systems, respectively, then $min\_card(\Delta|\mathbf{e}) \geq min\_card(\Delta'|\mathbf{e}\overrightarrow{\mathbf{e}})$, where $min\_card(\Delta|\mathbf{e})$ is the minimum cardinality of diagnoses represented by Δ under $\mathbf{e}$. If, however, $\mathbf{e}$ contains a complete instantiation of split variables then $min\_card(\Delta|\mathbf{e}) = min\_card(\Delta'|\mathbf{e}\overrightarrow{\mathbf{e}})$.

Therefore, to compute the exact minimum cardinality of diagnoses we perform branch-and-bound search in the space of instantiations of split variables $\mathbf{S}$: At each node of the search tree we compute $min\_card(\Delta'|\mathbf{e}\overrightarrow{\mathbf{e}}\mathbf{s}\overrightarrow{\mathbf{s}})$ by evaluating the DNNF compilation of Δ', where $\mathbf{e}$ is the given observation, and $\mathbf{s}$ is the current partial assignment to split variables. At leaf nodes of the search tree (where $\mathbf{s}$ is a complete assignment to split variables) the values of $min\_card(\Delta'|\mathbf{e}\overrightarrow{\mathbf{e}}\mathbf{s}\overrightarrow{\mathbf{s}})$ give candidate minimum cardinalities, and elsewhere they give lower bounds to prune the search. As soon as the lower bound becomes equal to or greater than the current minimum cardinality the branch gets pruned.

The efficiency of branch-and-bound search is significantly affected by the choice of a search *seed* (the starting upper bound on the minimum cardinality) as well as the ordering of the search variables and their values. We note that a good search seed can be computed from the given abnormal observation and the system model: We propagate the values of system inputs (given in the observation) into the model and compute normal values at the outputs of system components. We then compare the values of outputs of components with those given in the observation. If the values of k components are found to be inconsistent then it is intuitive to note that k is the upper bound on the minimum cardinality of diagnoses. Hence k can be used as a seed for the search.

For variable and value ordering of search variables, we use the *nogood*-based scoring heuristic similar to [Siddiqi and Huang, 2009]. Each value ($true$ or $false$) of a split variable X is associated with a score $S(X = x)$. The score $S(X)$ of the variable X is the average of the scores of its values. Once a variable X is assigned a value x, the amount of increase in the bound caused by it is added to the score of $X = x$. That is, the score $S(X = x)$ is updated as $S(X = x) = S(X = x) + (new\_bound - current\_bound)$, where $current\_bound$ is the bound before assigning the value x to X and new_bound is the bound after the assignment. We compute initial scores for each variable by setting each value of that variable hypothetically (separately) and computing the increase caused in the initial bound (i.e., the bound computed before starting the search) by that assignment.

During the search, the heuristic chooses a variable X with the highest score and assigns values to it in decreasing order of the scores of its values. Since, over the course of the search the scores can become misleading, as past updates to the scores may have lost their relevance under the current search conditions. Therefore, we divide the scores by a constant c periodically after having visited p search nodes. Empirically, we found that $p = 500$ and $c = 2$ worked well.

4.2 Search for Diagnoses

Once the minimum cardinality of diagnoses has been computed branch-and-bound search is again performed to compute the minimum-cardinality diagnoses. We discuss two strategies in this regard and then combine the two strategies to gather the advantages of both.

Search Strategies. In the first strategy, we note that if $\mathbf{e}$ contains a complete assignment to split variables then $min\_card\_diags(\Delta|\mathbf{e}) = min\_card\_diags(\Delta'|\mathbf{e}\overrightarrow{\mathbf{e}})$, where

Algorithm 1 BNB-MINC-DIAGS : Branch-and-Bound Search for minimum-cardinality diagnoses

procedure BNB-MINC-DIAGS ($\Delta', \mathbf{h}, \mathbf{s}$)
inputs: $\{\Delta'$: DNNF of split system$\}$, $\{\mathbf{h}$: partial assignment to health variables$\}$ $\{\mathbf{s}$: assignment to split variables$\}$
global variables: $\{\mathbf{e}$: observation$\}$, $\{\mathbf{H}$: health variables$\}$, $\{mincard$: minimum cardinality$\}$, $\{\mathbf{D}$: set of minimum-cardinality diagnoses$\}$

1: **if** (it is the root node of search ||
   $(\Delta'|\mathbf{he}\overrightarrow{\mathbf{e}}\mathbf{s}\overrightarrow{\mathbf{s}} == false)$ ||
   $card(\mathbf{h}) + min\_card(\Delta'|\mathbf{he}\overrightarrow{\mathbf{e}}\mathbf{s}\overrightarrow{\mathbf{s}}) > mincard$) **then**
2: $\quad$ $\mathbf{s}$ = search a complete assignment to split variables such that $\Delta'|\mathbf{he}\overrightarrow{\mathbf{e}}\mathbf{s}\overrightarrow{\mathbf{s}}$ is consistent and $card(\mathbf{h}) + min\_card(\Delta'|\mathbf{he}\overrightarrow{\mathbf{e}}\mathbf{s}\overrightarrow{\mathbf{s}}) \leq mincard$
3: $\quad$ **if** ($\mathbf{s}$ is not empty) **then**
4: $\quad\quad$ $\mathbf{P} = \{X : okX = false \in \mathbf{h}\}$
5: $\quad\quad$ $\mathbf{D} = \mathbf{D} \cup \{\{\mathbf{P}\} \times min\_card\_diags(\Delta'|\mathbf{he}\overrightarrow{\mathbf{e}}\mathbf{s}\overrightarrow{\mathbf{s}})\}$
6: $\quad$ **else**
7: $\quad\quad$ **return**
8: **if** $card(\mathbf{h}) == mincard$ **then**
9: $\quad$ **return**
10: pick some $okX \in \mathbf{H}$ such that $okX = okx \notin \mathbf{h}$
11: **for** each value okx_i of variable okX **do**
12: $\quad$ $\mathbf{h} \leftarrow \mathbf{h} \cup \{okX = okx_i\}$
13: $\quad$ BNB-MINC-DIAGS ($\Delta', \mathbf{h}, \mathbf{s}$)
14: $\quad$ $\mathbf{h} \leftarrow \mathbf{h} \backslash \{okX = okx_i\}$

$min\_card\_diags(\Delta|\mathbf{e})$ is the set of minimum-cardinality diagnoses represented by Δ under $\mathbf{e}$. Therefore, we search in the space of assignments to split variables and explore all those leaf nodes where the computed minimum cardinality is equal to the actual minimum cardinality. Hence a branch is pruned only if the lower bound exceeds the current minimum cardinality. At such leaf nodes we compute diagnoses as $min\_card\_diags(\Delta'|\mathbf{e}\overrightarrow{\mathbf{e}}\mathbf{s}\overrightarrow{\mathbf{s}})$. The union of all such sets of diagnoses gives us the required set of minimum-cardinality diagnoses. The advantage of this approach is that at a leaf node we can enumerate significantly large number of diagnoses from the DNNF. We noted that in several cases we had to explore extremely large search spaces while the number of diagnoses was reasonably small, and it seemed more useful to search in the space of assignments to health variables.

Therefore, in the second strategy we considered search in the space of assignments to health variables. Each partial assignment to health variables gives us a partial diagnosis. To enumerate all minimum-cardinality diagnoses we have to check each partial assignment to health variables for validity. As described earlier, the number of health variables as well as the diagnoses are often quite small due to abstraction, which gives advantage to this approach. However, for cases where the number of diagnoses was very large it proved quite inefficient as each and every diagnosis had to be searched.

Combined Approach. In the combined approach, we start by searching in the space of assignments to health variables. At each node of the search tree we check the current partial assignment $\mathbf{h}$ to health variables (current partial diagnosis) for *validity*, where $\mathbf{h}$ is *valid* if and only if it is *logically consistent* with the observation, and it can be extended to a minimum-cardinality diagnosis. For this purpose we perform another branch-and-bound search in the space of assignments to split variables and try to find a complete assignment $\mathbf{s}$ to these variables such that $\Delta'|\mathbf{he}\overrightarrow{\mathbf{e}}\mathbf{s}\overrightarrow{\mathbf{s}}$ is consistent (i.e., does not evaluate to $false$) and $card(\mathbf{h}) + min\_card(\Delta'|\mathbf{he}\overrightarrow{\mathbf{e}}\mathbf{s}\overrightarrow{\mathbf{s}}) \leq$ the minimum cardinality of diagnoses. If such an assignment $\mathbf{s}$ does not exist we regard $\mathbf{h}$ as *invalid* and backtrack. If $card(\mathbf{h})$ equals the minimum cardinality we backtrack as $\mathbf{h}$ need not be extended further.

At each node of the search tree where $\mathbf{h}$ is valid we compute all minimum-cardinality diagnoses from Δ' under $\mathbf{he}\overrightarrow{\mathbf{e}}\mathbf{s}\overrightarrow{\mathbf{s}}$ (the precise method of computing these diagnoses is given in Algorithm 1, lines $4-5$, to be described later). Note that if $\mathbf{B}$ is the set of components assumed broken in $\mathbf{h}$, then, intuitively, all computed diagnoses will also declare the components $\mathbf{B}$ as broken. Also the computed set of diagnoses will be a sub-set of the set of all minimum-cardinality diagnoses that declare the components $\mathbf{B}$ as broken.

The union of all the sets of diagnoses computed at each node of the search (where $\mathbf{h}$ is valid) gives us the required minimum-cardinality diagnoses of the system abstraction. A formal proof of the completeness of this procedure is omitted due to lack of space. However, the completeness follows from the fact that we are exhaustively searching in the space of assignments to health variables, and the techniques to avoid redundancy described below only avoid having to enumerate the same diagnoses multiple times.

Avoiding Redundancy. During the search the set of recorded diagnoses can be treated as *nogoods* to prune the search space of assignments to health variables. For this purpose we utilize a technique similar to *watch literals* (used in satisfiability): As soon as all but one components that appear in an already recorded diagnosis have been assumed broken in the current partial diagnosis then the remaining component in the recorded diagnosis is forced to be healthy.

Although nogoods help avoid some redundancy, several diagnoses will still be enumerated multiple times, which can lead to inefficiency. For example, suppose that the current partial diagnosis $\mathbf{h}$ is declared valid during search because we are able to find a complete assignment $\mathbf{s}$ to split variables without violating the above mentioned validity criteria for $\mathbf{h}$. Hence we enumerate a set of diagnoses $\mathbf{D}$ from Δ' under $\mathbf{he}\overrightarrow{\mathbf{e}}\mathbf{s}\overrightarrow{\mathbf{s}}$. Let $\mathbf{h_e}$ is a an extension of $\mathbf{h}$ ($\mathbf{h_e} \supset \mathbf{h}$) that is also valid and the search for a complete assignment to split variables to check the validity of $\mathbf{h_e}$ happens to find the same assignment $\mathbf{s}$ that was found for $\mathbf{h}$. Suppose that we enumerate a set of diagnoses $\mathbf{D_e}$ from Δ' under $\mathbf{h_e}\mathbf{e}\overrightarrow{\mathbf{e}}\mathbf{s}\overrightarrow{\mathbf{s}}$. It is evident that $\mathbf{D_e} \subseteq \mathbf{D}$ and diagnoses $\mathbf{D_e}$ will get enumerated twice. We avoid this redundancy as follow.

Let Q be a child node of a node P in the search tree in that Q is the immediate successor of P in the tree. Let $\mathbf{h}$ and $\mathbf{h_e}$ be the partial diagnoses at P and Q respectively. Let $\mathbf{s}$ be the complete assignment to split variables under which the validity conditions for $\mathbf{h}$ are met. To check the validity conditions for $\mathbf{h_e}$ at the node Q we first try to see if the $\mathbf{h_e}$ can be declared valid using the assignment $\mathbf{s}$ to split variables that was previously used at the parent node P of Q. If both conditions of validity are met we regard $\mathbf{h_e}$ as valid but do not enumerate diagnoses from Δ' under $\mathbf{h_e}\mathbf{e}\overrightarrow{\mathbf{e}}\mathbf{s}\overrightarrow{\mathbf{s}}$ (as described earlier these diagnoses would have already been enumerated). Oth-

erwise we perform search on split variables and try to find a fresh assignment to split variables that can satisfy the validity conditions for $\mathbf{h_e}$. If such an assignment exists only then we enumerate diagnoses.

Variable and Value Ordering. For variable and value ordering of health variables we use a similar scoring heuristic as in the first stage of search. However, this time scores are based upon the amount of search performed to validate the current partial diagnosis at a search node. The idea, intuitively, is to reduce the amount of search required for validation. We noted that as we try to reduce the overall amount of search required for validation the amount of search performed in space of assignments to health variables also reduced.

Therefore, we favor those health variables and their values that may result in less amount of search for validation. Suppose that a health variable okX is assigned a value okx and we have to explore p search nodes to validate the current partial diagnosis then the score of $okX = okx$ is $S(okX = okx) = S(okX = okx) + 1/p$ (scores are higher for smaller values of p). If the current partial diagnosis is validated by using the assignment $\mathbf{s}$ (to split variables) passed by the parent node (and no search is performed) then a hypothetical value of p is used which is half of the value used at the parent node (assuming that assigning value to a health variable would reduce the amount of search required to find $\mathbf{s}$ by half). The scores are reduced periodically as in stage 1, described above.

We compute initial scores of health variables using the failure probabilities of system components, as computed in [Siddiqi and Huang, 2010]. We assume that each component fails with equal prior probability of 0.1 and then compute the posterior probabilities under the given observation by evaluating and differentiating [Darwiche, 2003] the DNNF of the split system. Let f be the (posterior) failure probability of a component X. The initial score of $okX = true$ is set to 0, while the initial score of $okX = false$ is set to f. This effectively gives an initial ordering to health variables based upon the failure probabilities of the corresponding components.

The Algorithm. A basic procedure for the second stage of search is given in Algorithm 1 (BNB-MINC-DIAGS). It accepts as input the DNNF compilation Δ' of the split system, the current partial assignment $\mathbf{h}$ to health variables $\mathbf{H}$, and the assignment $\mathbf{s}$ to split variables $\mathbf{S}$ that is passed to the child search node by the parent node. $\mathbf{s}$ is empty for the root node of the search tree. The procedure updates the computed set of diagnoses in the global variable $\mathbf{D}$.

Algorithm 1 is a recursive procedure executed for each partial diagnosis $\mathbf{h}$ explored by the search. For each $\mathbf{h}$, we start by checking the validity of $\mathbf{h}$ at lines $1-3$ in two steps. Specifically we first check if $\mathbf{h}$ can be declared valid using the assignment $\mathbf{s}$ to split variables passed as parameter by the parent node. This is done using the if condition at line 1, which can be described as follows: The current partial diagnosis cannot be declared valid by the virtue of $\mathbf{s}$ passed as parameter to the procedure if either of the three conditions are $true$: (1) $\mathbf{s}$ is empty (i.e., the current node is the root of the search tree). (2) $\Delta'|\mathbf{h}\overrightarrow{\mathbf{e}}\mathbf{s}\overrightarrow{\mathbf{s}}$ evaluates to $false$ (i.e., $\mathbf{s}$ is not consistent with $\mathbf{h}$, model, and the observation). (3) $card(\mathbf{h}) + min\_card(\Delta'|\mathbf{h}\overrightarrow{\mathbf{e}}\mathbf{s}\overrightarrow{\mathbf{s}}) > mincard$ (i.e., the minimum-cardinality condition is violated).

If all of the three conditions described in the previous paragraph are $false$ then $\mathbf{h}$ is considered valid, in which case we jump to line 8 of the algorithm, and as described earlier we do not need to enumerate diagnoses from Δ' in this case. However, if either of the above three conditions is $true$ the validity of $\mathbf{h}$ is then checked through a branch-and-bound search (line 2) performed in the space of assignments to split variables. As the result of this search if $\mathbf{h}$ was valid we get a new assignment $\mathbf{s}$ to split variables which satisfies the conditions specified at line 2, in which case we enumerate the set of minimum-cardinality diagnoses from Δ' (lines $4-5$) and update them in $\mathbf{D}$. Otherwise if $\mathbf{h}$ was invalid then $\mathbf{s}$ is empty and the algorithm backtracks (line 7).

After the validity of $\mathbf{h}$ has been established we check if $card(\mathbf{h}) == mincard$ and if so we backtrack (lines $8-9$), as we do not want to enumerate diagnoses of cardinality greater than $mincard$. Otherwise we pick an un-assigned health variable $okX \in \mathbf{H}$ (line 10), and for each of its values okx_i append $\mathbf{h}$ with $okX = okx_i$ (lines $11-12$) and recursively call the procedure BNB-MINC-DIAGS with arguments Δ', updated $\mathbf{h}$, and updated $\mathbf{s}$ (at line 13) (note that $\mathbf{s}$ is passed to the children nodes of the current search node as a parameter to the procedure call at line 13). After the recursive call $okX = okx_i$ is erased from $\mathbf{h}$ (line 14).

4.3 Diagnosis of Cones

We must take care of two things before we can correctly diagnose a cone of a split system. First, while propagating normal values in the system followed by faulty values to compute observation at the inputs and output of a cone, whenever we assign a value to (the output of) a component we must assign the same value to all of its clones and propagate the values of clones as well. Second, splitting changes the depths of components in the new system compared with their depths in the original system. We must keep track of the original depths of components and use those depth values while ordering components in an abstract diagnosis.

For example, one minimum-cardinality diagnosis of the circuit in Figure 1 (right) is $\mathbf{D} = \{E, A\}$, which declares the cone E as faulty. To compute an abnormal observation at the inputs and output of E (a valuation of variables $\{P, \hat{B}, E\}$), we re-order gates in $\mathbf{D}$ as $\{A, E\}$ (gate A was originally deeper than E). We then propagate the circuit inputs into the model. When we compute a value 0 at B we assume the same for $\hat{B}$ and propagate the value of $\hat{B}$ also. Thus we get the valuation $P \wedge \neg\hat{B} \wedge \neg E$. We then propagate the faults declared in $\mathbf{D}$ one by one after flipping the values of A and E and get the required valuation $\alpha = P \wedge \hat{B} \wedge E$.

We have done an enhancement in the original abstraction-based technique that provided substantial performance improvement: We note that a cone can be diagnosed multiple times during the refinement of abstract diagnoses. Hence there is a possibility that a cone is diagnosed multiple times under the same observation. Therefore, we maintain a hash table of observations on a cone and save the corresponding diagnoses, and avoid diagnosing a cone a second time if the cone has already been diagnosed with the same observation.

We now give results of our empirical analysis.

circuit	gates	cases	HDIAG			DCAS				
			health vars	solved	time	treewidth	health vars	split vars	solved	time
c432	160	350	59	350	0.21	15	63	13	350	0.31
c499	202	400	58	400	0.12	15	58	11	400	0.20
c880	383	400	77	396	0.07	15	77	10	396	0.12
c1355	546	400	58	398	0.16	15	58	11	398	0.25
c1908	880	160	160	122	368.13	22	161	29	153	82.25
c2670	1193	160	167	145	176.17	12	223	33	160	3.15
c3540	1669	40	353	0	-	10	379	125	30	173.78
c5315	2307	40	385	0	-	16	388	66	39	52.34
c6288	2416	40	1456	0	-	20	1456	75	11	305.10
c7552	3512	40	545	0	-	16	564	76	35	260.93

Table 1: Experiments.

5 Experiments

We conducted experiments using ISCAS-85 benchmark circuits and compared the new approach, referred to as DCAS (Diagnosis by Compilation, Abstraction, and Search), with HDIAG [Siddiqi and Huang, 2007].

Test cases were generated randomly as in [Siddiqi and Huang, 2007]: For each circuit, we randomly generated a set of complete instantiations of inputs and outputs accordingly to the correct behavior of the circuit. We then randomly flipped k primary outputs, with k ranging from 1 to 8, in each instantiation to get an (abnormal) observation (the minimum cardinality of the diagnoses was often close to the number of flipped outputs), except for c432 for which the range is from 1 to 7 as it has only 7 outputs. For circuits upto $c1355$ we generated 50 test cases for each cardinality. For larger circuits we generated less number of test cases due to limited computational resources. Thus, for $c1908$ and $c2670$ we generated 20 test cases, and for $c3540$ to $c7552$ we generated 5 test cases for each cardinality, respectively.

All experiments were conducted on a set of eight Quad Core machines each running linux with 2.4 GHz Intel Xeon X3220 CPU and 2 GB of RAM. A memory limit of 768 MB and a time limit of 30 CPU minutes were imposed on each test case. Note that only two test cases were executed on a machine at one time due to limited amount of available RAM.

When splitting nodes the objective is to gain more efficiency in compilation with less number of split variables, as complexity of search can grow exponentially with increase in the number of split variables. We used the heuristic studied in [Choi *et al.*, 2007], which achieves the same objective. In addition, one has to choose a treewidth that achieves the optimal overall performance in terms of space and time requirements. We tried a range of treewidths for each circuit to find the one that gave the optimal performance. Splitting achieved significant simplification in the circuit and offset any disadvantage of increase in the abstraction size due to this.

The results of the experiments are given in Table 1. The first column gives the name of the circuit, the second gives the number of gates in the circuit, the third gives the total number of test cases used, and the fourth and fifth columns compare HDIAG and DCAS respectively. For HDIAG we report the number of health variables (or the abstraction size) in the model, the number of cases solved, and the average running time to solve those cases. For DCAS we also report the reduced treewidths of the circuits and the number of split variables. The times for DCAS includes times for node-splitting, compilation, and search.

HDIAG could not solve any circuit beyond $c2670$, while DCAS can solve most of the diagnostic cases on every circuit, except on $c6288$. Almost all failures occurred in those cases where the number of diagnoses was too large such that all diagnoses could not be enumerated within the given space, except on $c6288$ where many failures occurred due to very large search space resulting from very large number of health variables. Note that $c6288$ has a flat structure (having very large abstraction size) and lacks the kind of hierarchy that is vital for the success of our techniques.

On cases that can be solved by both systems, the running times of both are either comparable (upto $c1355$) or DCAS is significantly faster than HDIAG, notably in case of $c1908$ where DCAS is almost 4 times faster and on $c2670$ where it is almost two orders of magnitude faster than HDIAG.

6 Conclusion

We have presented a new tool referred to as DCAS which uses model relaxation, abstraction, and search to compute minimum-cardinality diagnoses of a faulty system. The new approach significantly advances the state of the art by solving non-trivial diagnostic cases on large systems that could not be solved previously, and is much more efficient on cases that can be solved by the previous technique.

References

[Choi *et al.*, 2007] Arthur Choi, Mark Chavira, and Adnan Darwiche. Node splitting: A scheme for generating upper bounds in Bayesian networks. In *UAI*, pages 57–66, 2007.

[Darwiche, 2001] Adnan Darwiche. Decomposable negation normal form. *Journal of the ACM*, 48(4):608–647, 2001.

[Darwiche, 2003] Adnan Darwiche. A differential approach to inference in Bayesian networks. *Journal of the ACM*, 50(3):280–305, 2003.

[Pipatsrisawat and Darwiche, 2007] Knot Pipatsrisawat and Adnan Darwiche. Clone: Solving weighted Max-SAT in a reduced search space. In *AI*, pages 223–233, 2007.

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

[Siddiqi and Huang, 2007] Sajjad Siddiqi and Jinbo Huang. Hierarchical diagnosis of multiple faults. In *IJCAI*, pages 581–586, 2007.

[Siddiqi and Huang, 2009] Sajjad Siddiqi and Jinbo Huang. Variable and value ordering for MPE search. In *IJCAI*, pages 1964–1969, 2009.

[Siddiqi and Huang, 2010] Sajjad Siddiqi and Jinbo Huang. New advances in sequential diagnosis. In *KR*, pages 17–25, 2010.