Generating Manifestations of Max-Fault Min-Cardinality Diagnoses

Alexander Feldman[1] and **Gregory Provan**[2] and **Arjan van Gemund**[1]
[1]Delft University of Technology
Faculty of Electrical Engineering, Mathematics and Computer Science
Mekelweg 4, 2628 CD, Delft, The Netherlands
Tel.: +31 15 2781935, Fax: +31 15 2786632, e-mail: {a.b.feldman,a.j.c.vangemund}@tudelft.nl
[2]University College Cork, Department of Computer Science, College Road, Cork, Ireland
Tel: +353 21 4901816, Fax: +353 21 4274390, e-mail: g.provan@cs.ucc.ie

Abstract

Computing test vectors that are optimized to isolate faults is an important area of diagnostics. The literature has focused on test vectors for single-fault diagnoses. This article generalizes this, addressing the problem of computing Max-Fault Min-Cardinality (MFMC) observation vectors in Model-Based Diagnosis (MBD), and proposing two algorithms for solving it. An MFMC observation vector is an observation that, for a given system description, results in the maximum number of faults in the minimum cardinality diagnosis. Computing MFMC observation vectors has application in testability analysis, MBD benchmarking, optimal sensor placement and other areas of model-based reasoning. We discuss the high computational complexity of the MFMC problem and introduce stochastic methods to reduce the solution complexity. The first method for computing MFMC is based on importance sampling while the second one is based on simulated annealing. Both algorithms lead to significant speed-up compared to exhaustive search and perform best for different classes of system descriptions. The algorithms described in this paper have been implemented and tested on a benchmark of combinatorial circuits.

Introduction

The problem of computing minimum cardinality diagnoses, given an observation and a system description, is central to Model-Based Diagnosis. Consider the inverse problem of computing an observation which distinguishes a specific number of faulty components. Computing observations that distinguish a single failing component is studied by Automatic Test Pattern Generation (ATPG) and dates back to (Roth 1966). The goal of ATPG is, then, to compute a *sequence* of test vectors that can distinguish every possible single fault in a device.

Single-fault ATPG has been extended to finding observation vectors leading to double faults (Hughes 1988) and to multiple faults (Kubiak & Fuchs 1991). All of these approaches have several drawbacks, including: (1) they are inherently suboptimal, i.e., they do not answer the question of what is the maximum number of faults distinguishable by a single test vector, (2) they suffer from very high computational complexity, and (3) they severely limit the class of system abstractions by using various simulation techniques.

Few papers address algorithms computing observation vectors that distinguish the *maximum* number of failing components in a system, e.g., (Abramovici 1981). One can devise a class of more general algorithms for this, based on techniques from MBD and abductive reasoning. This paper formalizes the problem of finding Max-Fault Min-Cardinality (MFMC) observation vectors and proposes two algorithms for solving it. These two methods are very efficient, given the existence of a fast MBD engine.

One of the advantages of the algorithms in this paper over the related *k-fault* ATPG algorithms is that they do not impose *any limitations* on the model (e.g., they do not require stuck-at modes or unlimited observability). This makes them applicable not only to testing but to a wider range of Model-Based Reasoning problems. The MFMC algorithms can be used for MBD benchmarking (Provan & Wang 2007), optimal sensor placement (Console, Picardi, & Ribaudo 2000) and other applications.

A summary of our contributions follows. This paper introduces the MFMC problem and two algorithms for computing MFMC observation vectors. The first one is based on importance sampling and the second one on simulated annealing. The two algorithms are empirically analyzed on a number of diagnostic models. Furthermore, we discover some properties of the MFMC search and reason about its computational complexity.

The rest of this paper is organized as follows. The section which comes after this introduction defines the basic MFMC framework. It is followed by a short discussion on some complexity issues. The fourth section suggests algorithms for solving the MFMC problem. Finally, we evaluate the empirical performance of the algorithms.

Preliminaries

The discussion starts by formalizing some basic notions in MBD, extending the notions in (de Kleer, Mackworth, & Reiter 1992). A *model* of an artifact is represented as a propositional **Wff** over some set of variables V. Discerning a subset of them as *assumable* or *observable* gives us a diagnostic system.

Definition 1 (Diagnostic System)**.** A diagnostic system DS is defined as the triple $\mathrm{DS} = \langle \mathrm{SD}, \mathrm{COMPS}, \mathrm{OBS} \rangle$, where SD is a propositional theory describing the behavior of the system, COMPS is a set of assumable variables in SD, and OBS is a set of some observable variables in SD.

Although it is not strictly necessary, throughout this paper we will assume that $\text{OBS} \cap \text{COMPS} = \emptyset$.

A Running Example

The simple Boolean circuit shown in Figure 1 is used to illustrate the notions in this papers and the workings of the algorithms we propose. The 2-to-4 line demultiplexer consists of four Boolean inverters and four and-gates.

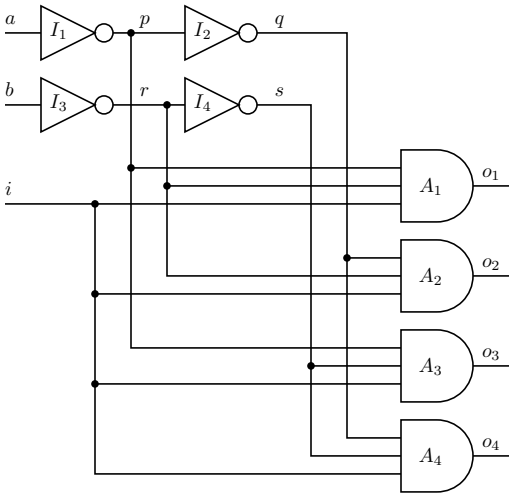

Figure 1: A 2-to-4 line demultiplexer.

The expression $h \Rightarrow (o \Leftrightarrow \neg i)$ models an inverter, where the variables i, o, and h represent input, output and health respectively. Similarly, an and-gate is modeled as $h \Rightarrow (o \Leftrightarrow i_1 \wedge i_2 \wedge i_3)$. These propositional formulae are copied for each gate in Figure 1 and their variables renamed in such a way as to properly connect the circuit and disambiguate the assumables, thus generating the following propositional formula for SD:

$$\text{SD} = \begin{cases} I_1 \Rightarrow (a \Leftrightarrow \neg p) \\ I_2 \Rightarrow (p \Leftrightarrow \neg q) \\ I_3 \Rightarrow (b \Leftrightarrow \neg r) \\ I_4 \Rightarrow (r \Leftrightarrow \neg s) \\ A_1 \Rightarrow (o_1 \Leftrightarrow i \wedge p \wedge r) \\ A_2 \Rightarrow (o_2 \Leftrightarrow i \wedge r \wedge q) \\ A_3 \Rightarrow (o_3 \Leftrightarrow i \wedge p \wedge s) \\ A_4 \Rightarrow (o_4 \Leftrightarrow i \wedge s \wedge q) \end{cases}$$

The set of component (assumable) variables is $\text{COMPS} = \{I_1, \ldots, I_4, A_1, \ldots, A_4\}$. The set of observable variables is $\text{OBS} = \{i, a, b, o_1, \ldots, o_4\}$.

Diagnostic Framework

The traditional query in MBD results in finding terms of assumable variables which are explanations for the system description and an observation. The first definition of diagnosis uses a set notation.

Definition 2 (Diagnosis)**.** A *diagnosis* for the system $\text{DS} = \langle \text{SD}, \text{COMPS}, \text{OBS} \rangle$, given an observation term α over variables in OBS, is a set $D \subseteq \text{COMPS}$ such that:

$$\text{SD} \wedge \alpha \wedge \left[\bigwedge_{c \in D} \neg h_c\right] \wedge \left[\bigwedge_{c \in (\text{COMPS} \setminus D)} h_c\right] \not\models \bot$$

Under *lex parsimoniae* we are interested in computing those diagnoses that are not subsumed by other diagnoses of $\text{SD} \wedge \alpha$.

Definition 3 (Minimal Diagnosis)**.** A diagnosis D is minimal iff no proper subset $D' \subset D$ exists, such that D' is also a diagnosis.

Throughout this paper we interchangeably use a propositional notation for expressing diagnoses. In this case we simply construct a conjunction of literals, each literal having a negative sign if its respective variable is in D and a positive sign otherwise. Consider the example from Figure 1 and an observation $\alpha = a \wedge b \wedge i \wedge \neg o_4$. In this case $D_1 = \{I_1, I_2\}$ is a diagnosis and $D_2 = \{I_1\}$ is a minimal diagnosis (there are four more minimal diagnoses for $\text{SD} \wedge \alpha$). Alternatively, instead of D_1 we may write $D'_1 = \neg I_1 \wedge \neg I_2 \wedge I_3 \wedge I_4 \wedge A_1 \wedge \ldots \wedge A_4$.

The *cardinality* of a diagnosis D is the size of D and is denoted as $|D|$. It represents the number of faulty components in COMPS given SD and α. Next to computing minimal diagnoses, it is of interest to MBD to compute some or all minimal-cardinality diagnoses, given a diagnostic problem.

Definition 4 (Minimal-Cardinality Diagnosis)**.** A diagnosis D is a minimal-cardinality diagnosis if it is a minimal diagnosis and no other diagnosis D' exists such that $|D| < |D'|$.

The cardinality of a minimal-cardinality diagnosis computed from a system description SD and an observation α is denoted as $\mathit{MinCard}(\text{SD} \wedge \alpha)$. For our example and the observation $\alpha = a \wedge b \wedge i \wedge \neg o_4$, it follows that $\mathit{MinCard}(\text{SD} \wedge \alpha) = 1$. Note that in this case all minimal diagnoses are also minimal-cardinality diagnoses.

There are minimal diagnoses which are not minimal-cardinality diagnoses. Let us consider, for example, the diagnostic system $\text{DS} = \langle \text{SD}, \text{COMPS}, \text{OBS} \rangle$, where $\text{SD} = (h_1 \wedge h_2 \wedge x) \vee (h_4 \wedge x)$, $\text{COMPS} = \{h_1, h_2, h_3, h_4\}$, $\text{OBS} = \{x\}$, and an observation $\alpha = x$. In this case, $D_1 = \{h_1, h_2, h_3\}$ is a non-minimal diagnosis, $D_2 = \{h_1, h_2\}$ and $D_3 = \{h_4\}$ are minimal diagnoses, but only D_3 is a minimal-cardinality diagnosis.

Definition 5 (MFMC Observation)**.** Given a system description SD, a Max-Fault Min-Cardinality (MFMC) observation is an instantiation α over variables in OBS such that $\mathit{MinCard}(\text{SD} \wedge \alpha)$ is maximized.

Throughout this paper we will use the term *MFMC diagnosis*, that is any of the minimal-cardinality diagnoses D for which α is an MFMC observation. The cardinality of the MFMC diagnosis of a diagnostic system DS is denoted as $\mathit{MaxCard}(\text{DS})$.

The MBD literature often considers a class of theories that define normative behavior of their components only, i.e., models which specify no fault-modes. These models are sometimes referred to as *weak-fault models*, or *ignorance of abnormal behavior* (de Kleer, Mackworth, & Reiter 1992), or, in our case, *implicit fault systems*.

Definition 6 (Implicit Fault System)**.** A diagnostic system DS belongs to the class **IFS** iff SD is in the form $(h_1 \Rightarrow F_1) \wedge \ldots \wedge (h_n \Rightarrow F_n)$ such that for $1 \leq i, j \leq n$, $\{h_i\} \subseteq$ COMPS, $F_j \in$ **Wff**, and none of h_i appears in F_j.

Traditionally diagnosis and minimal diagnosis are defined only in the context of implicit fault systems. Note that a diagnosis assigns values to all variables in COMPS.

A stronger notion of diagnosis exists for systems that impose no restriction on the propositional theory (e.g., strong-fault models). To introduce these types of diagnoses we will borrow the next definition from (Darwiche 1998).

Definition 7 (Consequence)**.** Given a diagnostic system $\mathrm{DS} = \langle \mathrm{SD}, \mathrm{OBS}, \mathrm{COMPS} \rangle$, and an observation α, the consequence of $\mathrm{SD} \wedge \alpha$, is a sentence $Cons(\mathrm{SD} \wedge \alpha)$, such that all its literals are in COMPS, $\mathrm{SD} \wedge \alpha \models Cons(\mathrm{SD} \wedge \alpha)$ and for any term β, $\mathrm{SD} \wedge \alpha \models \beta$ if $Cons(\mathrm{SD} \wedge \alpha) \models \beta$.

The next definition gives us another way to represent diagnosis and a more expressive explanation of $\mathrm{SD} \wedge \alpha$.

Definition 8 (Partial Diagnosis)**.** Given a diagnostic system $\mathrm{DS} = \langle \mathrm{SD}, \mathrm{OBS}, \mathrm{COMPS} \rangle$, and an observation α, a partial diagnosis ω is defined as a term over the set of assumable variables $h \in$ COMPS, such that $\omega \models Cons(\mathrm{SD} \wedge \alpha)$.

By distinguishing these partial diagnoses only, which are not contained in other implicants of $Cons(\mathrm{SD} \wedge \alpha)$, we get partial diagnoses which are minimal under subsumption. Computing these diagnoses, however, is strictly more difficult than computing the set of all minimal diagnoses.

Definition 9 (Kernel Diagnosis)**.** A partial diagnosis ω is a kernel diagnosis iff no partial diagnosis ω' exists, such that ω' is the conjunction of a proper subset of the literals in ω.

Consider again the example from Figure 1 and an observation $\alpha = a \wedge b \wedge i \wedge \neg o_4$. In this case $\omega_1 = \neg I_1 \wedge \neg I_2$ is a partial diagnosis and $\omega_2 = \neg I_1$ is a kernel diagnosis. These are similar to the results for diagnosis and minimal diagnosis, but consider changing the models of all inverters in SD from $h \Rightarrow (o \Leftrightarrow \neg i)$ to $[h \Rightarrow (o \Leftrightarrow \neg i)] \wedge (\neg h \Rightarrow \neg o)$. In the latter scenario, diagnoses and minimal diagnoses are not defined and $\omega_1 = I_1 \wedge \neg I_2 \wedge I_3$ is a kernel diagnosis. Note that, for example, $\omega_2 = \neg I_1 \wedge \ldots \wedge \neg I_4$ is not a partial diagnosis even though it contains a superset of the faulty components in ω_1.

The cardinality of a partial diagnosis ω, denoted as $Card(\omega)$, is defined as the number of negative literals in ω. In the above example, $Card(\omega_1) = 1$. In a similar way we derive minimal-cardinality partial diagnosis and MFMC partial diagnoses. It can be shown, that if SD $\in$ **IFS** the MFMC cardinalities would be the same for both kernel and minimal diagnoses.

Our methods for computing MFMC observation vectors rely on a diagnostic oracle. This oracle is supplied with a system description SD and a candidate observation α. Depending on the implementation of this oracle, our algorithms will compute MFMC observation vectors for either minimal diagnoses or partial diagnoses. For the rest of this paper, when it is clear from the context or from the model, we will drop the qualification of the diagnosis type.

Before we proceed with the algorithmic sections, we can truncate the search space for MFMC observation vectors by realizing that, for finding $MaxCard(\mathrm{DS})$, it is necessary to instantiate all observable variables in SD.

Proposition 1 (Observation Monotonicity)**.** *Given that* SD *is a system description and α and β are two observations such that $\alpha \supseteq \beta$ then it holds that $MinCard(\mathrm{SD} \wedge \alpha) \geq MinCard(\mathrm{SD} \wedge \beta)$.*

Proof. The proof comes directly from the definitions of diagnosis. We construct a system of Boolean equations B in the following manner. First, the propositional **Wff** in SD is converted to a Boolean equation in a straightforward manner and the latter is added to B. Second, for each literal $l_i \in \alpha$, an equation of the form $l_i = 1$ or $l_i = 0$ (depending on the polarity of l_i) is appended to B. A system of Boolean equations B' is constructed from SD and β in an analogous way. The solutions of B and B' are the implicants of $\mathrm{SD} \wedge \alpha$ and $\mathrm{SD} \wedge \beta$, respectively. Observe, that, due to the fact that $\alpha \supseteq \beta$, the equations in B' are a superset of these in B and both are over the same set of variables. But $S(B') \leq S(B)$, where $S(X)$ denotes the number of solutions in a system X. The above holds also when the solutions of B and B' are ordered according to their cardinality. Hence, if a diagnosis with a cardinality smaller than the smallest cardinality diagnosis in B' exists, it is in B. □

Next we discuss some computational complexity properties of generating MFMC observation vectors.

Complexity of the MFMC Problem

We have already seen that there is dependency between the observability of a model and the cardinality of the MFMC diagnosis (the average complexity of the problem). Before we continue our reasoning with the worst-case complexity of MFMC, we motivate the need of MFMC by noting that building diagnosable systems is expensive in terms of sensors (observables).

Assume that we have a system with k binary-valued sensors and n components, and that each component can be either faulty or healthy, i.e., the fault description does not define failure modes. We define a failure as an instantiation of fault modes, and a test as an instantiation of sensors.

There are 2^k test-vector settings (the test space Γ), and $2^n - 1$ possible fault combinations (the failure space Ω).

The ability to isolate all fault combinations (in which case the system is diagnosable) is typically impossible for practical reasons. In most cases, for a large system, it is too expensive to provide enough sensors to ensure full diagnosability. The following theorem defines the number of sensors needed to isolate q failures:

Theorem 1. *The minimum number of sensors needed to isolate q failures is given by $\lceil log_2(q+1) \rceil$.*

Proof. If we have q binary sensors, then there are 2^q distinct sensor signatures, of which $\{0, ..., 0\}$ is the nominal signature. Hence $2^q - 1$ signatures denote distinguished failures. This can be rearranged simply as follows:

$$
\begin{array}{rcl}
q \text{ sensors} & \rightsquigarrow & 2^q - 1 \text{ distinguished failures} \\
q+1 \text{ sensors} & \rightsquigarrow & 2^q \text{ distinguished failures} \\
\lceil log_2(q+1) \rceil \text{ sensors} & \rightsquigarrow & q \text{ distinguished failures}
\end{array}
$$

□

As a consequence, we must operate in a world where some failures are indistinguishable given Γ (they mask each other). In this article we choose to focus on the probabilistically most-likely failures, assuming that we have a probability distribution over the individual faults and all faults are mutually independent.

To complete our theoretical notions, we reason about the worst-case complexity of the MFMC problem. We can show that solving a simplified restriction of the MFMC problem is NP-complete.

Theorem 2 (Complexity of Restricted MFMC)**.** *Given a diagnostic model* $\mathrm{DS} = \langle \mathrm{SD}, \mathrm{OBS}, \mathrm{COMPS} \rangle$ *and a diagnosis D, it is NP-complete to determine a manifestation for the observation* α*.*

This theorem can be proven by observing that it is just the dual to the problem of determining the existence of a diagnosis D given the observation α (Theorem 4.7 of (Bylander *et al.* 1991).)

Further, the complexity of the MFMC problem is likely to be higher than that of isolating multiple-fault diagnoses (which is NP-complete (Bylander *et al.* 1991; Friedrich, Gottlob, & Nejdl 1990)), or that of computing a minimum-size test set to isolate all single stuck-at faults in electronic circuits (which is NP-complete (Krishnamurthy & Akers 1984)). With regard to Multiple-Fault Diagnosis (MFD), MFMC introduces an optimization task that makes multiple calls to an MFD oracle, which is clearly harder. With regard to testability analysis, MFMC addresses the multiple-fault case, and is applicable to arbitrary models, and not just stuck-at circuit models.

It is also likely that approximating MFMC within a constant factor is intractable. Approximating the *single-fault* minimum-size test set within a factor $\delta > 1$ of optimality is NP-hard (Krishnamurthy & Akers 1984).

As a consequence of intractability and other practical issues, such as dealing with failures which are indistinguishable (they mask each other), we focus on the probabilistically most-likely failures, assuming that we have a probability distribution over the individual faults and all faults are mutually independent.

Algorithms for Computing MFMC

In this section we discuss algorithms for computing MFMC. The first one is based on exhaustive search, hence it is suitable for understanding the basics of the MFMC computation only. The second algorithm borrows from Importance Sampling (IS) (Yuan & Druzdzel 2006) to skip over health assignment leading to faults of low cardinality. Finally, we suggest a simulated annealing algorithm for generation of MFMC.

A Naïve Brute-Force Algorithm

Obviously the model of the 2-to-4 line demultiplexer belongs to **IFS**. An algorithm which finds such an observation by trying all possible instantiation of the observable variables is shown in Algorithm 1.

Algorithm 1 An exhaustive search algorithm for generation of MFMC observation vectors.

```
1:  function NaïveMFMC(SD, OBS) returns a term
        inputs: SD, a propositional theory
                OBS, a set of observable variables
        local variables: α, ω, R, terms
                         M, an integer, initially 0
2:      for all α ← INSTANTIATE(OBS) do
3:          ω ← FINDMCDIAG(SD ∧ α)
4:          if M < COUNTFAULTS(ω) then
5:              R ← α
6:              M ← COUNTFAULTS(ω)
7:          end if
8:      end for
9:      return R
10: end function
```

The outer loop of Algorithm 1 tries all the $2^{|\mathrm{OBS}|}$ instantiations of the variables in OBS. For each possible instantiation α it finds the minimal cardinality diagnosis by issuing a call to the FINDMCDIAG subroutine. The observation leading to a diagnosis with a maximum number of faults (in this example COUNTFAULTS simply counts the number of negative assumable literals in ω) is preserved as a result.

Any method for computing a diagnosis can be used as an implementation of FINDMCDIAG and various methods like compilation (Darwiche 2001) or heuristics and conflict exploitation (Williams & Ragno 2004) can be used to speed-up this function.

For the circuit shown in Figure 1, Algorithm 1 exhaustively generates 128 instantiations of OBS. For example, consider the arbitrary assignment $\alpha = \neg a \wedge \neg b \wedge i \wedge \neg o_1 \wedge \ldots \wedge \neg o_4$. Clearly, the only minimal-cardinality diagnosis of $\mathrm{SD} \wedge \alpha$ is $\omega = \neg A_1$, and $Card(\omega) = 1$.

For the example demultiplexer, $MaxCard(\mathrm{SD}) = 4$ and there is a total of 4 observation vectors that can discern minimal-cardinality diagnosis of 4 faults. One of these max-fault min-cardinality observation vectors is $\alpha_{mfmc} = \neg a \wedge \neg b \wedge \neg i \wedge o_1 \wedge \ldots \wedge o_4$. Interestingly, from all the 128 possibilities, there are 8, 40, 53, and 23 observation vectors leading to a nominal, single-fault, double-fault and triple-fault diagnosis respectively.

Computing MFMC Approximations through Importance Sampling

Our IS algorithm is based on a weighted approach and works for a restricted set of system descriptions. Furthermore, the algorithm uses an extension to DS, a valuation function $Pr : \mathrm{COMPS} \rightarrow [0, 1]$. In practice, Pr assigns *a priori* probabilities to the system failure modes.

We assign an *a priori* valuation to an assignment α using $Pr(\alpha) = \Pi_{x \in \alpha \cup \mathrm{COMPS}} Pr(x)$, which corresponds to

assuming that all variables in SD are conditionally independent.

Consider systems that model the faulty behavior of their components. For some of them, it is possible to partition the set of observable variables OBS into two subsets IN and OUT (denoting input and output variables respectively), and after giving values to IN and COMPS, to use a reasoning algorithm (e.g., unit-propagation) to find a *unique assignment* to the values in OUT.

Definition 10 (Explicit Fault System)**.** Given a system DS and a partitioning $\text{OBS} = \text{IN} \cup \text{OUT}$, $\text{DS} \in \textbf{EFS}$ if for any instantiation ϕ of all variables in $\text{IN} \cup \text{COMPS}$, it holds that there is exactly one term ψ such that $\phi \models \text{SD} \wedge \psi$ and ψ is an instantiation of all variables in OUT.

The above restriction on the class of the propositional models allows us to introduce the algorithm which follows.

Algorithm 2 An algorithm for computing an MFMC approximations by using Importance Sampling.

```
1: function ISOBS(DS, IN, Pr, Pr*) returns a term
       inputs: DS = ⟨SD, COMPS, OBS⟩, a diag. system
               IN, a set of variables, IN ⊆ OBS
               Pr, a valuation function
               Pr*, a biasing pdf
       parameters: S#, integer, number of samples
       local variables: i, o, h, ω, R, terms
                        M, s, integers, initially 0
2:     while s < S# do
3:         ⟨h, i⟩ ← INSTANTIATE(COMPS, IN, Pr, Pr*)
4:         o ← PROPAGATE(SD ∧ i ∧ h)
5:         ω ← FINDMCDIAG(SD ∧ i ∧ o)
6:         if M < COUNTFAULTS(ω) then
7:             R ← i ∧ o
8:             M ← COUNTFAULTS(ω)
9:         end if
10:        s ← s + 1
11:    end while
12:    return R
13: end function
```

Algorithm 2 uses *simulation* to compute a subset of all (physically) possible states of a system. A biasing probability density function (pdf) is used to increase the probability of a sample being consistent with a higher number of faults. For each of the $S_{\#}$ samples over the input values i, the outputs o are computed and the minimal diagnosis consistent with $i \wedge o$ is computed. The observation which leads to a maximum number of faults is preserved throughout the sampling process and returned at the end of the procedure.

We discuss the implementation of the INSTANTIATE function which determines the quality of the observation vectors and the performance of the algorithm. Assuming equal probabilities for the input variables, it implements the function given next for assigning values to the variable set supplied as an argument.

$$P(x = \textbf{True}) = \begin{cases} k\,[1 - Pr(x)] & : \quad x \in \text{COMPS} \\ 0.5 & : \quad x \in \text{OBS} \end{cases}$$

The above function uses a skewing coefficient k to scale the probability of an assumable variable being instantiated as faulty. In general, this coefficient depends on the model and we will observe its effect in the experimentation section.

The implementation of the PROPAGATE subroutine is straightforward. We suggest the use of a Binary Constraint Propagation (BCP) method which can efficiently derive a satisfying assignment for the output variables. The auxiliary function GETOBS is used to discern these literals in a model of SD which instantiate observable variables.

Algorithm 2 computes minimal cardinality diagnosis by issuing a call to the FINDMCDIAG subroutine which is the same as in Algorithm 1. The observation leading to a diagnosis with a maximum number of faults (in this example COUNTFAULTS simply counts the number of negative assumable literals in ω) is preserved as a result. The number of calls to the potentially most expensive subroutine FINDMCDIAG decreases from $2^{|\text{OBS}|}$ (in the case of an exhaustive search) to $S_{\#}$, where $S_{\#}$ is generally low.

To illustrate the workings of Algorithm 2 on the 2-to-4 line demultiplexer we have introduced in the running example, it is necessary to change the system description used by the exhaustive search algorithm. The reason for this is that due to the weak-fault model the propagation routine would not be able to derive the values of the output variables given all inputs and health. In order to fix this, instead of one we use two assumable variables per logic-gate f_0 and f_1 to denote "stuck-at-zero" and "stuck-at-one" respectively.

The new formula for modeling an inverter is $(f_0 \Rightarrow \neg o) \wedge (f_1 \Rightarrow o) \wedge (\neg f_0 \wedge \neg f_1 \Rightarrow (\neg i \Leftrightarrow o)) \wedge (\neg f_o \vee \neg f_1)$. Each of the and-gates is represented as $(f_0 \Rightarrow \neg o) \wedge (f_1 \Rightarrow o) \wedge (\neg f_0 \wedge \neg f_1 \Rightarrow (i_1 \wedge i_2 \wedge i_3 \Leftrightarrow o)) \wedge (\neg f_o \vee \neg f_1)$. Again, we have to rename the variables for each gates, receiving a system containing eight Boolean equations. For brevity, we will omit the actual system description from this paper.

In running Algorithm 2 on the demultiplexer circuit, we assign the set of inputs $\text{IN} = \{a, b, i\}$, the set of output variables $\text{OUT} = \{o_1, o_2, o_3, o_4\}$, the Pdf $Pr(f_0 = \textbf{True}) = Pr(f_1 = \textbf{True}) = 0.01$ and $k = 25$. The number of samples $S_{\#}$ has been set to 25.

For the demultiplexer circuit, Algorithm 2 works as follows. First it draws random values with equal probabilities for the input variables a, b and i. Then it draws values for the "stuck-at-zero" and "stuck-at-one" assumables with probability 0.25. After propagation, the values for the output variables $o_1, \ldots, o_4$ are computed. At the end, from all samples the observation consistent with minimal-cardinality diagnosis of maximum number of faults is chosen. For this example run, let this observation be $\alpha_{is} = a \wedge \neg b \wedge \neg i \wedge o_1 \wedge \neg o_2 \wedge o_3 \wedge o_4$. As the reader can see, this is consistent with a minimal cardinality triple-fault diagnosis $\omega_{\text{is}} = \big(A_1 \equiv F^1\big) \wedge \big(A_3 \equiv F^1\big) \wedge \big(A_4 \equiv F^1\big)$, where the proposition $A_n \equiv F^1$ denotes an and-gate "stuck-at-one". The received observation leads to a diagnosis of acceptable quality, but the sampling continues until $S_{\#} = 25$. A higher number of samples increases the probability of finding an observation leading to a quadruple fault which is the optimum for this example.

A Simulated Annealing Algorithm

Algorithm 3 has no restrictions on the theories for which it can compute MFMC approximations. The technique it employs is simulated annealing (Rutenbar 1989).

Algorithm 3 A simulated annealing algorithm for generation of MC observation vectors of multiple faults.

```
1:  function MCHILLCLIMB(SD, OBS) returns a term
        inputs: SD, propositional theory
                OBS, set of observable variables
        parameters: T_min, real, "cool-off" temperature
                    T_max, real, starting temperature
                    N, integer, number of tries
                    r, real, decay rate
        local variables: v_c, v_n, terms
                         t, j, ΔE, integers
                         T, real, current temperature
2:      t ← 0
3:      repeat
4:          v_c ← INSTANTIATERANDOM(OBS)
5:          j ← 0
6:          repeat
7:              T = T_max e^{-jr}
8:              for all v_n ← FLIPOBSERVABLE(v_c) do
9:                  ΔE ← f(SD ∧ v_n) − f(SD ∧ v_c)
10:                 if ΔE > 0 then          ▷ Better MFMC?
11:                     v_c ← v_n           ▷ Accept the move.
12:                 else                    ▷ Consider going downhill.
13:                     if RAND() < e^{T^{-1} ΔE} then
14:                         v_c ← v_n
15:                     end if
16:                 end if
17:             end for
18:             j ← j + 1
19:         until T < T_min
20:         t ← t + 1                       ▷ Number of attempts.
21:     until t = N
22:     return v_c
23: end function
```

Algorithm 3 performs a maximum number of N independent attempts, each one starting from a random observation vector. These random observations are returned by INSTANTIATERANDOM, which assigns with equal probability **True** or **False** to each observable. As we will see in the experimentation section, a random observation vector is most likely to lead to a diagnosis of cardinality $M/2$, where M is the number of faults in the MFMC diagnosis.

The algorithm manipulates the initial random observation in an attempt of reaching a good optimum. The manipulation of the observation vector, aiming at "hill climbing" is performed by the FLIPOBSERVABLE subroutine. The idea is to try "flipping" variables in the observation vector until a "flip" leads to an improvement in the fault cardinality. In some of the cases, however, "flipping" the value of an observable will lead to a decrease in the associated number of faults. In these cases Algorithm 3 considers accepting the "worse" observation in its current state v_c with some probability depending on the current temperature T. This is to allow the search to "escape", if stuck in a local optimum.

The probability of accepting a state v_n which is worse than the current one in v_c, defines the process of "cooling", which gives the name of Algorithm 3. The temperature T, which starts from $T_{\max}$ and decreases gradually to $T_{\min}$, results in such "worse" states being accepted with higher probability in the beginning of each iteration and decreasing the likelihood of such "flips" towards the end of the search, i.e., when the search "freezes".

The "value flips" are repeated until the current observation in v_c becomes consistent with a minimal-diagnosis fault of improved cardinality, computed by the evaluation function f. The implementation of f returns the number of faults in the minimal cardinality diagnosis consistent with SD $\wedge$ α and is the same as in Algorithm 3. In particular it calls COUNTFAULTS and FINDMCCARD.

The parameters of the simulated annealing algorithm which affect its performance and the quality of the MFMC observation vectors are $T_{\min}$, $T_{\max}$, N, and r. These are the starting and "cool-off" temperatures, the number of "tries" and the decay rate, respectively. Similar to (Spears 1996), we will choose $r = (|\text{OBS}| * N)^{-1}$. The rationale behind this choice of r is that we would like a "faster" decaying when the problem size or the number of random restarts increase. Increasing the temperature range $T_{\max} - T_{\min}$ or reducing the decay rate r would allow more thorough search to be performed from each randomly chosen position.

The RAND function returns a normally distributed random number x such that $0 \leq x < 1$. In the beginning of the decaying process T is close to 1, hence the search is stochastic, hence more likely to escape local optima. In the cooling process the search becomes like an ordinary hill-climbing.

Experimental Results

This section presents an empirical analysis of our algorithms.

Implementation Notes and Test Set Description

Our implementation is approximately 1000 lines of C code (excluding the diagnosis computation) and is a part of the LYDIA[1] package. Two variants of the critical subroutine for finding a minimal-cardinality diagnosis have been used. These utilize conflict-based search (Williams & Ragno 2004) and exploitation of structure (Feldman & van Gemund 2006). Despite the above state-of-the-art implementations, the diagnosis search routine constrains the efficiency of the MFMC observation vector search, which is not surprising knowing that we are trying to diagnose circuits with observations consistent with multiple-faults of large cardinality.

Table 1 summarizes the benchmark we have used for testing of our algorithms. All the models are derived from the 74XXX family of arithmetic circuits.

[1]This package for model-based fault diagnosis can be downloaded from `http://fdir.org/lydia/`.

Name	Description	V_w	H_w	O	C_w	C_s
74180	9-bit parity check	38	14	12	48	90
74139	2-to-4 decoders	42	18	14	52	106
74153	4-to-1 selector	44	16	14	62	110
74182	4-bit CLA	47	19	14	75	132
74283	4-bit adder	89	40	14	130	250
74L85	4-bit comparator	93	41	14	134	257
74181	4-bit ALU	138	62	22	216	402

Table 1: Basic characteristics of the fault models from the 74XXX circuits family.

Some of the basic properties of the models we have used for benchmarking are shown in Table 1. For the models belonging to **IFS**, V_w, H_w, and O are the total number of variables, the number of assumables $|\mathrm{COMPS}|$ and the number of observables $|\mathrm{OBS}|$, respectively. For the strong-fault models, used in the testing of Algorithm 2, the number of assumables is $H_s = 2H_w$, the number of observable variables is the same as in the weak-fault models and the total number of variables is $V_s = V_w + H_w$. Columns C_w and C_s show the number of clauses in the CNF representations of the weak and strong system descriptions, respectively.

All the experiments described in this paper are performed on a host with 1.86 GHz Pentium M CPU and 2 Gb of RAM.

MFMC Vector Sizes and Performance Results

A series of exhaustive MFMC experiments allowed us to make an interesting observation. For all the benchmark circuits we have tested, the empirical pdf of the MFMC fault cardinalities, in respect to the observation vectors, approximates a binomial pdf within a very small margin. The results for two of the circuits are plotted in Figure 2.

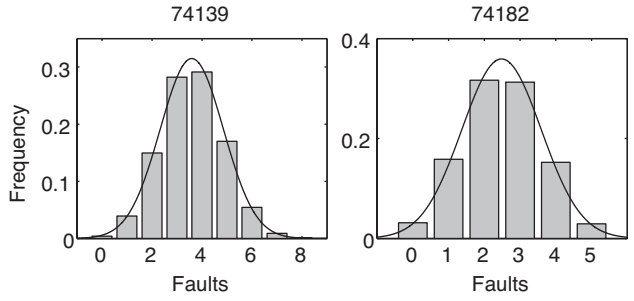

Figure 2: Empirical distributions of the fault cardinalities and normal pdf lines.

In the 74182 circuit, for example, given a randomly generated observation, the probabilities of a nominal kernel diagnosis or a quintuple fault are equal and very small. Analyzing the reasons for this behavior is a topic of its own, but our suggestion is that the underlying cause is the uncertainty introduced by the limited observability of the circuits (observability is defined as the ratio between the number of observable variables and all model variables).

The two implementations we test in this section are parameterized as follows. The importance sampling algorithm uses biasing coefficient $k = 25$ and the number of samples is equal to the number of components in the strong-fault model, i.e., $S_{\#} = H_s$. For the simulated annealing, we have set $T_{\min} = 0.1$, $T_{\max} = 0.105$, and $N = 4$.

Name	T_e	M	T_i	M_i	T_s	M_s
74180	1.4	2	0.04	2	0.1	2
74139	13.5	8	0.08	4.9	0.6	7.4
74153	8.8	2	0.08	2	0.2	2
74182	53.4	5	0.23	4.4	2.7	5
74283	371.9	5	5.74	3.9	18	3.9
74L85	196.9	3	3.28	3	14.5	3
74181	–	–	596.48	5.4	6114.4	6.5

Table 2: Fault cardinalities and wall-clock times [s] for computing of MFMC observation vectors.

The wall-clock times for the importance sampling and simulated annealing searches are shown in columns T_i and T_s of Table 2, respectively. The cardinalities of the MFMC observation vectors, computed by the two algorithms, are in M_i and M_s. As both MFMC computation methods in this paper are randomized, we have averaged the values of T_i, M_i, T_s, and M_s over 10 runs. It was possible to perform an exhaustive search, in all the test cases except for 74181. The results are shown in columns M and T_e, the former denoting the cardinality of the global MFMC optima and the latter – the computation time.

It is visible from Table 2 that Algorithm 2 outperforms Algorithm 3 by a factor of $2.5 - 11.7$. The cause of this is mainly in the smaller number of diagnoses which have to be computed, i.e., the importance sampling is more informed in reaching a local optimum. The observation vectors computed by Algorithm 2 lead to diagnoses of somewhat smaller cardinality than these computed by Algorithm 3. The difference is usually one fault, except in the 74139 circuit.

Figure 3 shows the progress of the MFMC search for two of the benchmark models. We note that for the 74139 circuit, the local optimum is found in the third iteration, while 74182 reaches its result from the first attempt. As a result decreasing N would keep the cardinality of the result but decrease the number of diagnostic computations.

From Figure 3 it is also visible that it is possible to climb to a good local optimum from an arbitrary initial random instantiation. This justifies the "observation bit flipping" operator (implemented in the FLIPOBSERVABLE subroutine of Algorithm 3) for climbing uphill in the stochastic search.

Conclusion

We have described two methods for computing MFMC observation vectors. The first algorithm, based on importance sampling, is applicable to a subset of all possible propositional theories, in particular to strong-fault models. This restriction, which does not exist in the simulated annealing

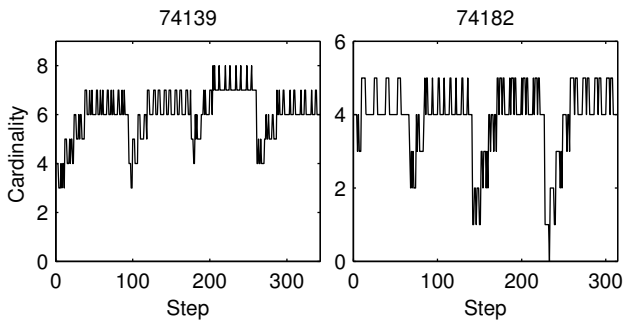

Figure 3: Fault cardinalities during sample simulated annealing sessions.

algorithm, results in a smaller number of calls to the underlying diagnostic engine and faster diagnostic reasoning.

We have studied the real-world behavior of the two algorithms on a series of combinatorial circuits. In all experiments, the number of faults in the diagnostic results was close to the global optimum: in some cases Algorithm 3 leads to diagnoses having one more fault than those computed by Algorithm 2.

An MBD oracle has been used in both of the algorithms. The only disadvantage of this is the underlying complexity of the diagnostic algorithms. Existing MBD heuristic methods, however, are tailored towards testing candidate diagnoses in order of likelihood. With the further development of the reasoning techniques, like the ones discussed in this paper, we expect new MBD heuristic methods to be developed that benefit from focusing the diagnostic search on high-cardinality diagnoses.

Finding MFMC observation vectors is of significant practical importance, and we expect more attention in the model-based reasoning community. Finally, we hope that the MFMC search will improve the algorithms for MBD, which in their turn will allow us to compute MFMC observations of better cardinality and for bigger models.

Acknowledgments

This work has been supported by STW grant DES.7015 and SFI grant 04/IN3/I524.

References

Abramovici, M. 1981. A maximal resolution guided-probe testing algorithm. In *Proc. DAC'81*, 189–195.

Bylander, T.; Allemang, D.; Tanner, M.; and Josephson, J. 1991. The computational complexity of abduction. *Artificial Intelligence* 49:25–60.

Console, L.; Picardi, C.; and Ribaudo, M. 2000. Diagnosis and diagnosability analysis using PEPA. In *Proc. ECAI'00*, 131–135.

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

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

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

Feldman, A., and van Gemund, A. 2006. A two-step hierarchical algorithm for model-based diagnosis. In *Proc. AAAI'06*.

Friedrich, G.; Gottlob, G.; and Nejdl, W. 1990. Physical impossibility instead of fault models. In *Proc. AAAI*, 331–336.

Hughes, J. 1988. Multiple fault detection using single fault test sets. *IEEETCAD* 7(1):100–108.

Krishnamurthy, B., and Akers, S. B. 1984. On the complexity of estimating the size of a test set. *IEEE Trans. Computers* 33(8):750–753.

Kubiak, K., and Fuchs, W. K. 1991. Multiple-fault simulation and coverage of deterministic single-fault test sets. In *Proc. of the IEEE International Test Conference on Test*, 956–962.

Provan, G., and Wang, J. 2007. Automated benchmark model generators for model-based diagnostic inference. In *Proc. IJCAI'07*, 513–518.

Roth, J. P. 1966. Diagnosis of automata failures: A calculus and a method. *IBM Journal of Research and Development* 10:278–291.

Rutenbar, R. A. 1989. Simulated annealing algorithms: An overview. *IEEE Circuits and Devices* 5(1):19–26.

Spears, W. M. 1996. Simulated annealing for hard satisfiability problems. In *Second DIMACS Implementation Challenge: Cliques, Coloring and Satisfiability*, volume 26, 533–558.

Williams, B., and Ragno, R. 2004. Conflict-directed A* and its role in model-based embedded systems. *Journal of Discrete Applied Mathematics*.

Yuan, C., and Druzdzel, M. J. 2006. Importance sampling algorithms for Bayesian networks: Principles and performance. *Mathematical and Computer Modelling* 43(9-10):1189–1207.