On the Relation Between Intermittent Faults and Behavior Models

Roni Stern and **Meir Kalech** and **Ester Lazebnik**
Ben Gurion University of the Negev
Be'er Sheva, Israel

Abstract

Sequential diagnosis is a well-known type of the diagnosis problem in which multiple observations of a system are given as input such that some of these observations exhibit abnormal behavior. The task in sequential diagnosis is to find the set of components that are faulty and have caused the abnormal behavior. An additional challenge in sequential diagnosis is that some components may fail intermittently, i.e., behaves abnormally in one observation and behave normally in another. Most past work on sequential diagnosis applied a conflict-directed approach, where conflicts are extracted from each observation and minimal hitting sets of all conflicts are considered as diagnoses. Inspired by recent success in SAT-based approaches for solving classical diagnosis problems, we study in this work two ways to solve the sequential diagnoses by compiling it to Boolean satisfiability (SAT) and using state-of-the-art SAT solvers.

Introduction

The goal in a diagnosis problem is to find the set of components that are faulty and have caused the abnormal behavior. *Sequential diagnosis* is a type of diagnosis problem in which multiple observations are given, such that the observation were taken in different time steps. Some work on sequential diagnosis focus on how to actively generate these additional observations, e.g., by placing probes to provide additional measurements (de Kleer and Williams 1989; Siddiqi and Huang 2011; Feldman et al. 2013) or by actively testing the system on different sets of inputs and outputs (Feldman, Provan, and van Gemund 2010; Zamir, Stern, and Kalech 2014). In this work we use the term *sequential diagnosis* to refer only to the diagnostic challenge of how to infer diagnoses from multiple observations, regardless of how these observations were generated.

Indeed, the problem of finding diagnoses for a set of observations poses additional challenges over the classical diagnosis problem (in which a single observation is given). First, the sequential diagnosis problem has a higher complexity, as it needs to reason about more information. Second, a faulty component may behave normally in some observations, thus making it more difficult to find. Such faults, which manifest only in some observations, are called *intermittent faults*.

We focus on a model-based approach for solving the sequential diagnosis problem, i.e., we assume that a model of the system's behavior is given and infer diagnoses from the model and the observations. Several approaches were previously proposed for model-based sequential diagnosis. For example, some proposed a conflict-directed approach (Raiman et al. 1991; De Kleer 2009), generating conflicts for each observation and merging them together. Others proposed a hierarchical approach based structural abstraction and compilation to d-DNNF (Siddiqi and Huang 2011). Encouraged by recent success of model-based diagnosis algorithms for classical diagnosis that are based on compilation to Boolean satisfiability (SAT) (Metodi et al. 2014; Marques-Silva et al. 2015), we investigate in this work a SAT-based approach for sequential diagnosis.

Two SAT-based algorithms are presented: `one-SAT` and `divide-and-join`. The former compiles the problem to a single formula, and the latter compiles each observation to its own SAT formula and then joins the resulting diagnoses. Importantly, we do not presume to claim that the SAT-based algorithms are better or worse than the algorithms from the other approaches (conflict-directed, d-DNNF). Our goal is to propose, analyze, and evaluate SAT-based solutions for the sequential diagnosis problem.

The paper is structured as follows. First, background and formal definition of model-based diagnosis and sequential model-based diagnosis is given. Then, we characterize different assumptions about the behavior of components in a sequential diagnosis problem. Then, we introduce the two SAT-based algorithms we propose and evaluate them empirically. Lastly, we conclude and discuss future work.

Model-Based Diagnosis

An MBD problem is defined by a tuple $\langle SD, COMPS, OBS \rangle$, where SD is a model of the diagnosed system, $COMPS$ is the set of system com-

ponents, and *OBS* is the observed behavior of the system (e.g., the observed inputs and outputs of the system). The system model *SD* specifies the normal behavior of every component $c \in COMPS$. We say that a component $c \in COMPS$ has a strong fault-model (SFM) if *SD* contains some information about how c behaves when faulty. Otherwise we say that c has a weak fault-model (WFM).

A solution to an MBD problem is a diagnosis. A diagnosis is an assumption about the behavior of the system components that is a plausible explanation of the observed behavior. To define a diagnosis formally, we introduce the following notation. Every component c is associated with health variable h_c. The domain of h_c, denoted $dom(h_c)$, is the set of possible assumptions about the behavior of c. These are called the behavior modes of c. The behavior modes of every component c consists a healthy mode, representing that c is assumed to be healthy, and one or more *fault modes*, representing various ways in which c can behave abnormally. We denote the health mode and the set of fault modes of component c by ok and F_c, respectively. Note that $dom(h_c) = \{ok\} \cup F_c$ and if c has a WFM then F_c contains a single fault mode *unknown*. A *health assignment* is an assignment of behavior modes to all the health variables. In logical terms, a health assignment is a conjunction of propositional literals, where each of these literals is of the form $h_c = m$ where $m \in dom(h_c)$. Finally, we can formally define a *diagnosis*.

Definition 1 (Diagnosis)**.** *A health assignment ω is a diagnosis iff $SD \wedge OBS \wedge \omega$ is satisfiable.*

The goal of an MBD algorithm is to return one or more diagnoses, as defined in Definition 1.

The number of diagnoses can be very large and there is therefore a need to prioritize them. A common method to do so is by using the notions of *subset-minimal* (SM) diagnoses and *minimal-cardinality* (MC) diagnoses, which are explained next. For a diagnosis ω we denote by ω^- the components in ω assigned to a faulty behavior mode, and the size of ω^- is referred to as the *cardinality* of ω, denoted by $|\omega|$. A diagnosis is a SM diagnosis iff there is no other diagnosis ω' such that $\omega'^- \subset \omega^-$. A diagnosis is a MC diagnosis iff there is no other ω' such that $|\omega'| < |\omega|$. Preferring SM diagnoses over diagnosis that are not SM, and preferring MC diagnoses over SM diagnoses, roughly corresponds to an Occam's Razor reasoning in which a simpler explanation – that which assumes less faulty components – is to be preferred over the more complex one. In this work we mostly focus on the problem of finding MC diagnoses.

Sequential MBD

An implicit assumption in the MBD problem definition above is that *OBS* is the observed system behavior at a specific point in time. It is often the case, however, that the diagnosing agent observes the system over time. This gives rise to the *sequential* MBD problem, which is an extended version of the MBD problem in which multiple observations are given as input. Formally, the sequential MBD problem is defined by a tuple $\langle SD, COMPS, T, \{OBS_t\}_{t \in T} \rangle$, where *SD* and *COMPS* are the system description and set of components as above, $T = \{t_1, \ldots, t_n\}$ is the set of time points in which the system was observed, and OBS_{t_i} is the observation at time point t_i. Following de Kleer (Raiman et al. 1991), we use the term *observation time* to refer to a point in time in which the system is observed, i.e., every $t \in T$ is an observation time.

A solution to a sequential MBD is also a diagnosis. However, a diagnosis in the context of sequential MBD is an assumption about the behavior of the system components that is a plausible explanation for *all* the observations. To formally define a diagnosis in this context, we define for every component c and observation time t a *timed health variable* $h_{c,t}$. The domain of $h_{c,t}$ is the behavior modes of component c and its value represents the behavior mode of c at observation time t. A timed health assignment is an assignment of behavior modes to all the timed health variables.

Definition 2 (Diagnosis for a sequential MBD problem)**.** *A health assignment ω is a diagnosis of a sequential MBD problem $\pi = \langle SD, COMPS, T, \{OBS_t\}_{t \in T} \rangle$ iff there exists a timed health assignment τ such that for every component c it holds that if $h_c = ok$ then $\forall t \in Th_{c,t} = ok$ and if $h_c \neq ok$ then $\exists t \in Th_{c,t} = h_c$*

As was the case in the classical MBD problem, the number of diagnoses for a sequential MBD problem can be very large, and therefore it is desirable to have some sort of minimality criteria over sequential MBD diagnoses to focus the problem solver. To this end, we define the cardinality of a sequential MBD diagnosis as the number of components that are assumed faulty in at least one of the observation times. Formally, the cardinality of a diagnosis ω, denoted by $|\omega|$ is given by

$$|\omega| = |\{c | c \in COMPS \text{ and } \bigvee_{t \in T} h(c,t) \neq ok\}|$$

Our goal in this work is to find MC diagnoses for the sequential MBD problem.

Intermittent and Non-Intermittent Behavior Modes

Definition 2 embodies a fundamental difference between MBD and sequential MBD: in general, a component may output different values for the same input values in different observation times. This can be due to changes in its state between observation times, and due to some un-modeled aspect of the environment. This is referred to as an *intermittent* behavior mode. However, it is sometimes reasonable to assume that the components behavior is consistent over time, i.e., that for the same input the component will generate the same output. This is referred to as a *non-intermittent* behavior mode.

As an example, software components are notorious for their intermittent behavior, due to the computer's multi-tasking nature and dependence on many external aspects (e.g., network speed). By contrast, it is reasonable to assume a non-intermittent behavior from a valve in a hydraulic system, since a leaking valve is expected to always leak when it is flooded with fluid.

We follow de Kleer (Raiman et al. 1991) and formally define a component with a non-intermittent behavior as follows. Let $in(c,t)$ and $out(c,t)$ be the values inputted to and outputted by a component c at observation time t.

Definition 3 (Non-Intermittent Behavior)**.** *A component c is said to have a non-intermittent behavior mode iff there exists a function F such that*

$$\forall t \in T : F(in(c,t)) = out(c,t)$$

A component with an intermittent behavior mode is a component for which such a function F may not exist.

The Relation Between Fault Modes and Intermittency

One can assume that a component has an intermittent (Int) or a non-intermittent (NotInt) behavior mode, and one can assume that a component has a strong fault model (SFM) or a weak fault model (WFM). The first observation we make in this work is that each of the resulting four combinations (Int+SFM, NotInt+SFM, Int+WFM, and NotInt+WFM) is possible. Next, we formally define each of these combinations.

Int+WFM This is the least constrained assumption one can have on a component. There is no constraints on the abnormal behavior of the component and the component may behave differently in different observation times. Let ϕ_c be a function describing the healthy behavior of component c, and let $h(c,t)$ be a predicate that is true iff component c follows its healthy behavior at observation time t, i.e., $h(c,t) \equiv (h_{c,t} = ok)$. A component that is Int+WFM is defined as follows:

$$\forall t \in T : h(c,t) \rightarrow (out(c,t) = \phi_c(in(c,t))) \quad (1)$$

NotInt+WFM The behavior of a component that is NonInt+WFM was described by de Kleer (Raiman et al. 1991). Although the faulty component's behavior is not specified, a NonInt+WFM component's faulty behavior is constrained to be consistent along observations (Definition 3). Thus, the formal definition of a NonInt+WFM component is a mixture of Definition 3 and Equation 1.

$$\exists \phi_c^F \ s.t. \ \forall t \in T : h(c,t) \rightarrow (out(c,t) = \phi_c(in(c,t))) \\ \wedge \neg h(c,t) \rightarrow (out(c,t) = \phi_c^F(in(c,t))) \quad (2)$$

In plain words, a NonInt+WFM component is a component that has a fault model and it is non-intermittent, but we have no knowledge about that fault model.

Int+SFM Here, we know the possible way in which the component may behave when faulty (the behavior modes). However, the component may act differently in different observation times, i.e., switch between a healthy behavior and each of the behavior modes. [1] Let $M_c = \{m_1, \ldots, m_{|M|}\}$ be the set of possible behavior modes of component c, and let $\phi_c^{m_i}$ denote the function describing the behavior of component c when in mode m_i. $f_i(c,t)$ is a predicate that is true iff component c is following the behavior of mode m_i at observation time t. An Int+SFM component is defined as follows:

$$\forall t \in T : h(c,t) \rightarrow (out(c,t) = \phi_c(in(c,t))) \\ \bigwedge_{m_i \in M} f_i(c,t) \rightarrow (out(c,t) = \phi_c^{m_i}(in(c,t))) \quad (3)$$

In addition, we must define that a component c at time t has exactly one behavior mode – either healthy or one of the fault modes.

NotInt+SFM This is the most constrained behavior mode. A NonInt+SFM component must behave consistently throughout the observations, and its faulty behavior is specified as one of the fault modes in M. The formal definition is the same as in the Int+SFM case (Equation 3), but with an additional constraint to represent the non-intermittent behavior:

$$\forall t, t' \in T : h(c,t) = h(c,t') \\ \bigwedge_{m_i \in M} f_i(c,t) = f_i(c,t') \quad (4)$$

Finding Diagnoses

In this section we propose algorithms for solving a sequential diagnosis problem. Most of the presentation below as well as the experimental results focus on the Int+WFM configuration. We provide a brief discussion of how to extend these methods to the other configurations later in the paper.

Raiman et al. (1991) proposed a conflict-directed approach for solving the sequential MBD problem. In this approach, *conflicts* are extracted from each observation, and every minimal hitting set of all the conflicts is considered to be a diagnosis.[2]

In this work we explore a different approach to solving the sequential MBD problem, which is based on compiling the problem to Boolean satisfiability (SAT). The motivation for proposing such an approach is the recent success of SAT-based solvers for the classical,

[1] Here we describe a strong form of intermittency, where a component may have different fault modes in different observation times. One can also envision a weaker form of intermittency, in which a faulty component is associated with a single fault behavior mode, but can act normally in some observations. Addressing this is variant is left for future work.

[2] Verifying that a hitting set is a diagnosis can be done by running a SAT solver on each observation.

	WFM	SFM
Int.	Faulty behavior is unconstrained	Must follow a behavior mode but mode can differ between observations
Non-Int.	Not constrained by faulty behavior modes but must be consistent across observations	Must follow a single behavior mode across all observations

Table 1: A summary of a components faulty behavior in the different configurations of WFM/SFM and Int./NonInt.

single-observation, MBD problem (Metodi et al. 2014). Therefore, we first briefly describe the SAT-based approach for solving the classical MBD problem, and then explain two ways to extend it for solving the sequential MBD problem.

SAT-based MBD algorithm

A SAT solver is an algorithm that accepts as input a Boolean formula and outputs a satisfying assignment of that formula, if such exists, or false otherwise. It is straightforward to compile an MBD problem to a SAT problem, i.e., to a Boolean formula. We define clauses for every component specifying that every component has exactly one behavior mode, and an additional clause for every pair of component and behavior mode, specifying the component's behavior when in that behavior mode. In a WFM, we can define a single clause for a component, specifying that if it is healthy then it will act normally. In addition, there is a clause for every observation, specifying the values that were observed. The variables of this Boolean formula include the health variables h_c, and the values of these variables in a satisfying assignment are exactly a diagnosis.

To find MC diagnoses we add additional clauses to ensure that no more than UB health variables, where UB is an upper bound to the size of the MC diagnoses. There are standard ways to define such a cardinality constraint in a SAT formula (Bailleux and Boufkhad 2003; Silva and Lynce 2007). The process of finding MC diagnoses starts with finding the cardinality of the MC diagnoses. This is done by setting UB to an upperbound on the MC, and then iteratively decreasing UB until the resulting formula is not satisfiable, indicating that the previous UB is the minimal cardinality. Then, UB is set to that cardinality and a SAT solver is used to return all satisfying assignments, which are exactly all the MC diagnoses. For more details on this process see Metodi et al. (2014).

Sequential MBD as a Single Boolean Formula

The first way we propose to encode the sequential MBD problem is to construct a single Boolean formula that encodes the knowledge from all the observations. Doing this is under the assumption that components fail intermittently (recall that we focus on the Int+WFM) is fairly simple, since the observations can be encoded independently to a Boolean formula and we can just merge them together. Note that all the variables that represent the internal state of the system must be duplicated, to allow them to receive different values for different observations.

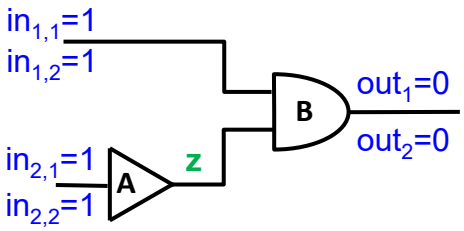

Figure 1: A simple example of a system with two observations.

To illustrate this, consider the simple example depicted in Figure 1. The inputs $in_{1,1}$, $in_{1,2}$, $in_{2,1}$, and $in_{2,2}$ are the first and second input of the first and second observation, such that $in_{i,j}$ is the i^{th} input of the j^{th} observation. Similarly, out_1 and out_2 are the outputs of the first and second observations, respectively. The variable z represents the value of an internal state of the system – the output of component A. As explained above, the variables for the internal state of the system are duplicated and hence in the Boolean formula for Figure 1 we have z_1 and z_2, which are the values of the output of component A at observation times 1 and 2, respectively. The resulting Boolean formula is given below: Equations 5-8 describe the normal behavior of components A and B, Equation 9 describes the observations' inputs, and Equation 10 describes the observations' outputs.

$$(h_{A,1} = ok) \rightarrow (z_1 = 1 - in_{2,1}) \quad (5)$$
$$\wedge (h_{B,1} = ok) \rightarrow ((z_1 \wedge in_{1,1}) = out_1) \quad (6)$$
$$\wedge (h_{A,2} = ok) \rightarrow (z_2 = 1 - in_{2,2}) \quad (7)$$
$$\wedge (h_{B,2} = ok) \rightarrow ((z_2 \wedge in_{1,2}) = out_2) \quad (8)$$
$$\wedge in_{1,1} = 1 \wedge in_{1,2} = 1 \wedge in_{2,1} = 1 \wedge in_{2,2} = 1 \quad (9)$$
$$\wedge out_1 = 1 \wedge out_2 = 1 \quad (10)$$

To find MC diagnoses, we also add a set of clauses to constrain the cardinality of the returned diagnoses. This is done by defining a health predicate H_c that is not timed (in contrast to the time-health variable $h_{c,t}$) for every component c, such that H_c is true iff c is assumed to behave normally in all observation times. Thus, each such health predicate is associated with the following

clause:

$$H_c \leftrightarrow \bigwedge_{t \in T} (h_{c,t} = ok)$$

For the example in Figure 1 we add the following clauses for health predicates H_A and H_B.

$$H_A \leftrightarrow ((h_{A,1} = ok) \wedge (h_{A,2} = ok)) \quad (11)$$

$$H_B \leftrightarrow ((h_{B,1} = ok) \wedge (h_{B,2} = ok)) \quad (12)$$

The process of finding all MC diagnoses is similar to the process described earlier for the classical MBD problem. A cardinality constraint is added to the Boolean formula that constrains these number of health predicates that are set to false. The constraint starts with some upper bound UB on the cardinality of the MC diagnoses, and we decrease UB iteratively (one by one) until the resulting Boolean formula is not satisfiable. This indicates that the previous value of UB is the minimal cardinality, and we set the cardinality constraint to this value to find all MC diagnoses. We call the above algorithm, which compiles the sequential MBD problem into a single Boolean formula, the `one-SAT` algorithm.

Joining Diagnoses of Different Observation

Encoding the knowledge about all observations into a single Boolean formula allows using the full power of modern SAT solvers. However, the size of the resulting encoding grows linearly with the number of observations. This can become a big computational problem since the worst case runtime complexity of modern complete SAT solvers is exponential in the size of the encoding (due to the P vs. NP issue). Next, we propose an approach that solves each observation independently, and then seeks to join the resulting set of diagnoses into a single diagnosis for the entire sequential MBD problem.

Let $\Pi = \langle SD, COMPS, T, \{OBS_t\}_{t \in T} \rangle$ be a sequential MBD problem. We define $\Pi_i = \langle SD, COMPS, OBS_i \rangle$ as the classical MBD problem that uses the same system model and components as Π but considers only observation i. Let $\Omega(\Pi)$ and $\Omega(\Pi_i)$ denote the set of all diagnoses for the sequential and classic MBD problems Π and Π_i, respectively. Since we focus on WFM a component in a diagnosis is either assumed to be healthy (ok) or not. Thus, we can represent a diagnosis as the set of components that are assumed to be faulty instead of a health assignment (which maps every component – healthy and faulty – to its behavior mode). For convenience of notation, we do so hereinafter, and thus every element in $\Omega(\Pi)$ and $\Omega(\Pi_i)$ is simple a set of components. Slightly abusing standard relational algebra terminology, we define the *join* operation between two sets of diagnoses Ω_i and Ω_j, denoted $\Omega_i \bowtie \Omega_j$, as follows:

$$\Omega_i \bowtie \Omega_j = \{\omega_i \cup \omega_j | \omega_i \in \Omega_i, \omega_j \in \Omega_j\}$$

Since we assume that faults are intermittent, then

$$\Omega(\Pi) = \Omega(\Pi_1) \bowtie \Omega(\Pi_2) \bowtie \ldots \bowtie \Omega(\Pi_n) \quad (13)$$

Therefore, we can find all diagnoses for the sequential MBD problem Π by solving the classical MBD problems $\Pi_1, \ldots \Pi_n$ individually, and joining the results. Concretely, instead of trying to solve the large Boolean formula described in the previous section, we can solve n smaller Boolean formulas and join their results. We call this algorithm `divide-and-join`.

Recall that the worst case runtime of current complete SAT solvers is exponential in the size of the Boolean formula they are given. Thus, solving the n Boolean formulas that represents the MBD problems $\Pi_1, \ldots, \Pi_n$ has a worst case runtime that is exponentially smaller than solving the single Boolean formula for Π, as the encoding for Π is n times larger than that of Π_i.

However, there is no free lunch, and the runtime of `divide-and-join` can be as bad and even worse than the runtime of the `one-SAT` algorithm. This is because the runtime of the join operation. Under a naive implementation, the join operation requires running over the cross product of all the diagnoses sets $\Omega(\Pi_1), \ldots, \Omega(\Pi_n)$, thus requiring runtime that is exponential in the number of observations. It is not obvious, at least to the authors, if there is a more efficient way to compute this join (notice its difference from standard relational algebra join, which can be implemented more efficiently). Moreover, the number of diagnoses returned by each MBD problem Π_i can be exponential in the number of components. Lastly, each activation of the SAT solver incurs some overhead, which is incurred n times for `divide-and-join` (as it requires n activations of a SAT solver) while this overhead is only incurred once for `one-SAT`. Indeed, as we show in the experimental results, there is no universal winner and which algorithm is more efficient depends on various domain properties (see discussion later in the results section).

Finding Minimal Diagnoses

The `divide-and-join` algorithm was described above for finding all diagnoses, which, as discussed earlier, can be prohibitively large. The question we address next is whether the `divide-and-join` can be applicable for finding all SM diagnoses, and for finding all MC diagnoses.

It turns out that using `divide-and-join` for finding all SM diagnoses is straightforward, since the relation between all the diagnoses of Π and all the diagnoses of $\Pi_1, \ldots, \Pi_n$ is maintained also for finding all SM diagnoses. That is, if $\Omega^{SM}(\Pi)$ and $\Omega^{SM}(\Pi_i)$ are the set of SM diagnoses for Π and Π_i, respectively, then:

$$\Omega^{SM}(\Pi) = \Omega(\Pi_1)^{SM} \bowtie \Omega(\Pi_2)^{SM} \bowtie \ldots \bowtie \Omega(\Pi_n)^{SM} \quad (14)$$

Therefore, `divide-and-join` can be used as-is for finding SM diagnoses: simply find all SM diagnoses for $\Pi_1, \ldots, \Pi_n$ and join the results.

However, Equation 13 does not carry over for finding MC diagnoses. That is, the join of all MC diagnoses for $\Pi_1, \ldots, \Pi_n$ may not contain all MC diagnoses for Π and may contain diagnoses that are not MC diagnoses of Π.

Formally, if $\Omega^{MC}(\Pi)$ and $\Omega^{MC}(\Pi_i)$ are the set of SM diagnoses for Π and Π_i, respectively, then:

$$\Omega^{MC}(\Pi) \neq \Omega(\Pi_1)^{MC} \bowtie \Omega(\Pi_2)^{MC} \bowtie \ldots \bowtie \Omega(\Pi_n)^{MC} \quad (15)$$

As an example, consider a sequential MBD problem with two observations, such that:

$$\Omega^{SM}(\Pi_1) = \{\{A, B\}\}$$
$$\Omega^{SM}(\Pi_2) = \{\{E, F\}, \{A, B, C\}\}$$

Therefore

$$\Omega^{MC}(\Pi_1) = \{\{A, B\}\} \text{ and } \Omega^{MC}(\Pi_2) = \{\{E, F\}\}$$

The join $\Omega^{MC}(\Pi_1)$ and $\Omega^{MC}(\Pi_2)$ is the diagnosis $\omega = \{A, B, E, F\}$ with cardinality 4. However, the MC diagnosis for Π is actually $\omega' = \{A, B, C\}$ with cardinality 3. Therefore, `divide-and-join` cannot find MC diagnoses without further adaptations.

`Divide-and-Join` for Finding MC Diagnoses

Next, we describe how `divide-and-join` can be modified to find MC diagnoses. We call this algorithm `Divide-and-Join-MC`. To properly explain `divide-and-join-MC`, we require the following terminology. Let mc and mc_i be the cardinality of the MC diagnoses for the MBD problems Π and Π_i, respectively. Also, let $\Omega^n(\Pi)$ and $\Omega^n(\Pi_i)$ be the set of all SM diagnoses of cardinality n or less for Π and Π_i, respectively. A *cardinality bounds vector* is an n-ary vector $\mathbf{b} = \langle b_1, \ldots b_n \rangle$ such that for every i in the range $[1, n]$ it holds that $b_i \geq mc_i$.

We say that a cardinality bounds vector $\mathbf{b}$ is *MC-sufficient* iff

$$\Omega^{MC}(\Pi) \subseteq \Omega^{b_1}(\Pi_1) \bowtie, \ldots, \bowtie \Omega^{b_n}(\Pi_n)$$

Finding an MC-sufficient cardinality bounds vector is important since it proving that all MC diagnoses of Π have been found. Finally, for a set of diagnoses X, we define $MC(X)$ as the cardinality of the diagnoses that has the smallest cardinality in X, i.e., $MC(X) = \min_{\omega \in X} |\omega|$.

`Divide-and-Join-MC` starts by initializing a cardinality bounds vector $\mathbf{b}$ by the cardinality of the MC diagnoses of the individual MBD problems, i.e., initially $\mathbf{b} = \langle mc_1, \ldots, mc_n \rangle$. In every iteration, the algorithm computes for every problem Π_i all the SM diagnoses of cardinality equal to or lower than b_i. Then, the algorithm attempts to prove that $\mathbf{b}$ is an MC-sufficient cardinality bounds vector, by inspecting whether the join of these sets of diagnoses is the set of all MC diagnoses of Π. If the algorithm is able to prove that $\mathbf{b}$ is an MC-sufficient cardinality bounds vector - it terminates, returning all diagnoses in the join that have the smallest cardinality. Otherwise, the cardinality bounds vector is incremented (adding one to all its elements), and the process continues.

The key question is how to prove that a cardinality bounds vector is MC-sufficient. For this we provide the following simple rule: if UB is an upper bound on the MC of Π then $\mathbf{b} = \langle UB, \ldots, UB \rangle$ is MC-sufficient. The challenge is how to find a low enough UB. For every cardinality bounds vector $\mathbf{b}$, For every cardinality bounds vector $\mathbf{b}$, it holds that every diagnosis ω in $\Omega^{b_1}(\Pi_1) \bowtie, \ldots, \bowtie \Omega^{b_n}(\Pi_n)$ is a diagnosis of Π and thus $|\omega|$ is an upper bound on mc. Thus, every iteration of the algorithm can provide a better UB. Concretely, the `divide-and-join` algorithm starts with a cardinality bounds vector $\mathbf{b} = \langle mc_1, \ldots, mc_n \rangle$ and in every iteration obtains the current UB (by computing all diagnoses for $\mathbf{b}$ and joining them). If $UB = \min_i b_i$ all MC diagnoses have been found. Otherwise, a new iteration starts with every element in $\mathbf{b}$ incremented by one.

Name	$\lvert$COMPS$\rvert$	in	out
74181	65	14	8
74283	36	9	5
c432	160	36	7
c880	383	60	26

Table 2: The Benchmark suite: systems 74XXX and ISCAS-85.

Empirical Evaluation

We evaluated the proposed SAT-based algorithms – `one-SAT` and `divide-and-join` – on Boolean circuit systems from the 74XXX (Hansen, Yalcin, and Hayes 1999) and ISCAS-85 (Brglez, Bryan, and Kozminski 1989) standard benchmarks suites. The details (number of components and number of inputs and outputs) of the chosen systems are given in Table 2. To the best of our knowledge, there is no standard set of observations for sequential MBD problems. Thus, we generated for each of these systems random sequential MBD problems with 1, 2, 4, 6, 8, and 10 observations. Each sequential MBD problem was generated as follows. First, faults are injected to 2-4 components (components are selected randomly). Then, in every observation we generate random input values and propagate them in the system. The faulty components behave abnormally – i.e., negate the normal output – with probability $p_{int} \in \{0.3, 0.5, 0.7, 0.85, 1\}$, where p_{int} is a parameter which we varied. This parameter (p_{int}) controls the "intermittency" of the components, where $p_{int} = 0$ means faulty component always behave normally, while $p_{int} = 1$ means that the faulty component will behave abnormally in every observation. We generated 15 different sequential MBD problems for each configuration of system, number of faults, number of observations, and p_{int}.

We solved each problem with `one-SAT` and with `divide-and-join`, and measured the runtime required to find the first MC diagnosis, and the runtime required to find all MC diagnoses. All these values are presented in Figure 2, which shows runtime in seconds (y-axis) for different systems, ordered by increasing size. The results show several interesting trends. First, the runtime

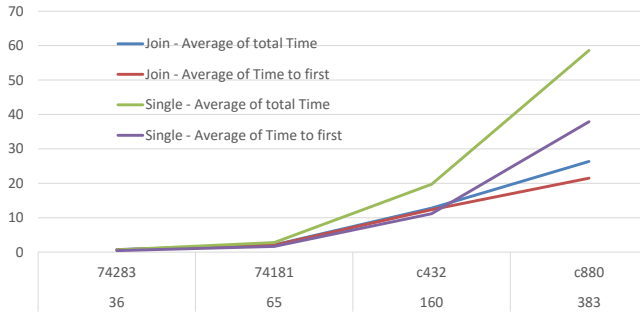

Figure 2: The runtime of one-SAT and divide-and-join as a function of the size of the system.

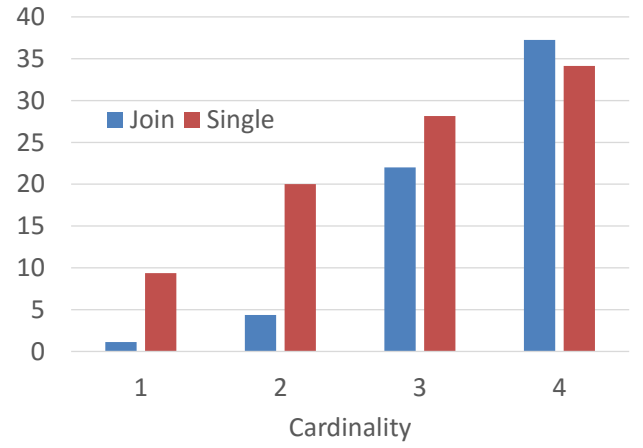

Figure 3: The runtime of `one-SAT` and `divide-and-join` as a function of the cardinality of the diagnosis.

of `one-SAT` becomes significantly larger than the runtime of `divide-and-join` for larger systems, highlighting the benefit of using `divide-and-join` for the harder problems. Second, observe that the time gap between finding the first diagnosis and all diagnoses is significant in `one-SAT` but less so in `divide-and-join`. This is due to the differences between how these algorithm works: the `one-SAT` algorithm first searches for a single MC diagnosis and then asks the SAT solver to find all other MC diagnoses. In contrast, the `divide-and-join` algorithm computes *all* MC diagnoses for each observations (and possible more, when increasing the cardinality bounds vector) when searching for the first MC diagnoses. Thus, the extra work the `divide-and-join` algorithm needs to do between finding a first MC diagnoses and finding all of them is smaller than in the `one-SAT` algorithm, and consequently the gap between finding a single MC diagnoses and a finding all of them is smaller. In the results below, we focus on the runtime of finding all MC diagnoses.

The cardinality of the diagnosis is an important factor in model-based diagnosis since the larger the cardinality the harder solving the diagnosis problem. In

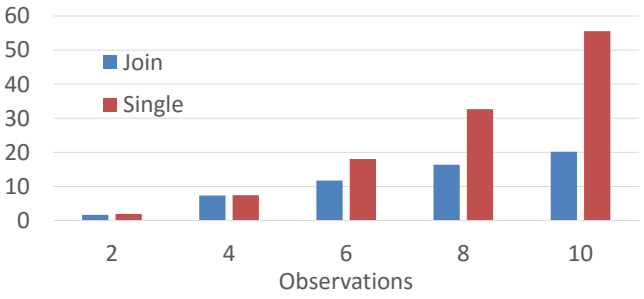

Figure 4: The runtime of Single and Join as a function of the cardinality of the diagnosis.

Figure 3 we can see the runtime of the `one-SAT` and `divide-and-join` algorithms as a function of the cardinality. It is interesting to see that the runtime growth of the `divide-and-join` algorithm is faster than that of `one-SAT`. `Divide-and-Join` is even slower than `one-SAT` for cardinality 4. This is reasonable since higher cardinality suggests that more observations exhibited abnormal behavior and therefore the join operation, done by the `divide-and-join` algorithm, will be more time consuming. By contrast, when the cardinality is small this suggests that some of the observations will not even exhibit abnormal behavior. The `divide-and-join` algorithm is especially suited to identify such cases, as these observations will not contribute any diagnosis and the join operation can simply skip them.

Lastly, we consider the impact of the number of observations on the runtime of the competing algorithms. The number of observations is a key factor in sequential MBD and considering more observations is expected to result in higher running time. Figure 4 shows the runtime of both algorithms for problems with different number of observations. We can see that in small number of observations there is no significant difference between the two algorithms and that indeed the runtimes of both algorithms increase with the number of observations. However, as the number of observations grows the growth in runtime of `divide-and-join` is more moderate than that of `one-SAT`. Thus, `divide-and-join` is more robust to increasing the number of observations. This can be explained by the fact that any added observation automatically incurs an increase in the size of the Boolean formula generated by `one-SAT`, and consequentially added runtime. This is not the case in `divide-and-join`, which is not affected, per se, by the number of observations, but by the number of diagnoses each observation has.

To conclude, both `divide-and-join` and `one-SAT` have configurations in which they are better: `divide-and-join` is faster for large systems and many observations but `one-SAT` is better for problems with high cardinality.

Conclusion and Future Work

In this work we proposed two SAT-based algorithms for solving sequential MBD problems. The first, named `one-SAT`, generated a single Boolean formula such that every satisfying assignment to that formulate is a diagnosis for the sequential MBD problem. The second SAT-based algorithm, named `divide-and-join`, generated a Boolean formula for every observation, and then merges the resulting diagnoses. There is no dominant algorithm, and we investigated empirically under which conditions each algorithm is superior.

This work only focused on the Int.+WFM setting. However, there are three other configurations that we did not deal with in this work: NonInt.+WFM, Int.+SFM, and NonInt.+SFM. The `one-SAT` algorithm can be easily adapted to Int.+SFM, and NonInt.+SFM configurations. Encoding non-intermittent components with WFM is more challenging, as it requires defining that a faulty behavior is consistent without any notion of what the faulty behavior will be. This, as well as adapting `divide-and-join` to these three configuration, is a topic for future work.

An important direction for future work is to consider a different objective than finding MC diagnoses. For example, some components may fail more often than others, and thus diagnoses that assume these components are faulty should be prioritized. Assigning a probability to a diagnosis of a sequential MBD problem is challenging when faulty components fail intermittently, as it requires estimating both the probability of a component to be faulty as well as the probability that a faulty component will behave abnormally. However, relatively recent work by de Kleer (2009) proposed a possible solution for this, and we intend to study how to incorporate this in our SAT-based solver.

References

Bailleux, O., and Boufkhad, Y. 2003. Efficient CNF encoding of boolean cardinality constraints. In *CP*, 108–122.

Brglez, F.; Bryan, D.; and Kozminski, K. 1989. Combinatorial profiles of sequential benchmark circuits. In *IEEE International Symposium on Circuits and Systems*, 1929–1934.

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

De Kleer, J. 2009. Diagnosing multiple persistent and intermittent faults. In *IJCAI*, 733–738.

Feldman, A.; de Castro, H. V.; van Gemund, A.; and Provan, G. 2013. Model-based diagnostic decision-support system for satellites. In *Aerospace Conference, 2013 IEEE*, 1–14. IEEE.

Feldman, A.; Provan, G.; and van Gemund, A. 2010. A model-based active testing approach to sequential diagnosis. *Journal of Artificial Intelligence Research (JAIR)* 39:301.

Hansen, M. C.; Yalcin, H.; and Hayes, J. P. 1999. Unveiling the ISCAS-85 benchmarks: A case study in reverse engineering. *IEEE Des. Test* 16:72–80.

Marques-Silva, J.; Janota, M.; Ignatiev, A.; and Morgado, A. 2015. Efficient model based diagnosis with maximum satisfiability. In *IJCAI*.

Metodi, A.; Stern, R.; Kalech, M.; and Codish, M. 2014. A novel sat-based approach to model based diagnosis. *Journal of Artificial Intelligence Research* 377–411.

Raiman, O.; de Kleer, J.; Saraswat, V. A.; and Shirley, M. 1991. Characterizing non-intermittent faults. In *AAAI*, 849–854.

Siddiqi, S. A., and Huang, J. 2011. Sequential diagnosis by abstraction. *Journal of Artificial Intelligence Research* 329–365.

Silva, J., and Lynce, I. 2007. Towards robust CNF encodings of cardinality constraints. In *CP*, 483–497.

Zamir, T.; Stern, R.; and Kalech, M. 2014. Using model-based diagnosis to improve software testing. In *Conference on Artificial Intelligence (AAAI)*, 1135–1141.