A Decentralised Symbolic Diagnosis Approach[1]

Anika Schumann[2] and **Yannick Pencolé**[3] and **Sylvie Thiébaux**[4]

Abstract.
This paper considers the diagnosis of large discrete-event systems consisting of many components. The problem is to determine, on-line, all failures and states that explain a given sequence of observations. Several model-based diagnosis approaches deal with this problem but they usually have either poor time performance or result in space explosion. Recent work has shown that both problems can be tackled when encoding diagnosis approaches symbolically by means of binary decision diagrams. This paper further improves upon these results and presents a decentralised symbolic diagnosis method that computes the diagnosis information for each component off-line and then combines them on-line. Experimental results show that our method provides significant improvements over existing approaches.

1 Introduction

This paper addresses the problem of diagnosing large component-based event-driven systems. Specifically, we are concerned with the on-line identification of all faults that explain the flow of observations continuously received from the system.

To assist operators in charge of the supervision of large and complex infrastructure networks in transport, energy, and telecommunication, there is a pressing need for model-based diagnosis methods that scale up. However, existing model-based approaches suffer from poor time performance or space explosion. This includes simulation-based approaches, which, such as that of Baroni *et al.* [1], start from a decentralised model of the system (a model of the individual components and their interactions), and track the possible system behaviours on-line as observations become available; the reliance on a decentralised model makes them space efficient, but the set of possible behaviours is so large that on-line computation can be time inefficient. At the other end of the spectrum, consider diagnoser based approaches, which, such as that of Sampath *et al.* [8] compile, off-line, a centralised system model into another finite state machine (the diagnoser) which efficiently maps observations to possible failures; here the space required by the centralised model, let alone that required by the diagnoser, constitutes a major problem.

Approaches with a better space-time tradeoff include the decentralised diagnoser approach [6] which precomputes diagnosers for small subsystems only, but needs to ensure consistency of the local diagnoses at run-time. Another line of work deals with the incremental on-line compilation of diagnosis information and its reuse [4]. Finally, symbolic representations, by means of binary decision diagrams (BDDs) have been applied to increase time and space efficiency of diagnosis methods [9, 11, 10, 5].

In this paper we present a symbolic decentralised diagnosis approach that further improves upon existing work. Instead of encoding one of the known decentralised methods symbolically, we define a new approach that directly exploits the advantages of BDDs. It is well known that BDDs can be used to efficiently encode and manipulate large state and transition sets. However, as demonstrated in [10], updating the set of possible faults upon reception of a new observation involves operations that cannot be efficiently performed by means of BDDs. Therefore, our decentralised diagnosis approach precomputes the diagnosis information off-line for each component. On-line we then only need to synchronise states and transitions to obtain the global diagnosis information.

Our experimental evaluation demonstrates that our approach can be much faster than symbolic simulation-based approaches. Perhaps more surprisingly, we show that its efficiency is comparable to that of centralised diagnosis approaches which have much higher space requirements.

The paper is organised as follows. We start with a short introduction to the diagnosis problem in discrete event systems and then present our decentralised diagnosis approach in Sections 3 and 4. Next, we present the symbolic encoding of our decentralised diagnosis model which we then use in Section 6 to retrieve the globally consistent diagnosis information on-line. Section 7 provides the experimental evaluation of our approach and Section 8 concludes with some words on related and future work.

2 Diagnosis Problem for Discrete Event Systems

The diagnosis problem for discrete-event systems consists in determining all system states and faults that are consistent with a sequence of observations. The set of consistent states and faults is known as the diagnosis information. As in [8], we assume that the diagnosed system is composed of a set of n individual components (finite state machines) $G_i = \langle X_i, \Sigma_i, x_{0_i}, T_i \rangle$ where X_i is the set of states, Σ_i the set of events, x_{0_i} the initial state, and T_i the transition set. The events are partitioned into observable Σ_{o_i} and unobservable Σ_{u_i} events, the latter of which is further partitioned into faults Σ_{f_i} and shared Σ_{s_i} events. The shared events are used to describe the communication between components. If a shared event is triggered in one component it is simultaneously triggered in all other components that define it. Figure 1 depicts simplified component models of a telecommunication application to which we will refer throughout the paper. The global model $G = \langle X, \Sigma, x_0, T \rangle$ of a system corresponds to the synchronous product of its component models. In order to simplify the definitions that follow, we introduce the concept of event paths.

[1] Work was done in the framework of NICTA's SuperCom project. NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

[2] Cork Constraint Computation Centre, University College Cork, Ireland, email: a.schumann@4c.ucc.ie. The author is currently funded by the Science Foundation Ireland under the ITOBO Grant No. 05/IN/I886.

[3] LAAS-CNRS, University of Toulouse, France, email: Yannick.Pencole@laas.fr

[4] Australian National University and NICTA, Australia, email: Sylvie.Thiebaux@anu.edu.au

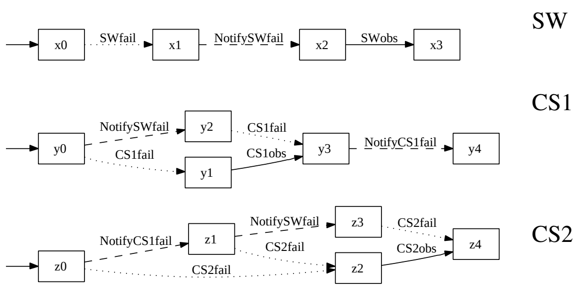


Figure 1. Simplified component models for a switch SW and two control stations $CS1$ and $CS2$. Solid lines denote observable transitions, dashed lines shared transitions and dotted lines failure transitions.

Definition 1 (Event Paths) *Let $FSM = \langle X, \Sigma, x_0, T \rangle$ denote a finite state machine. An event path $P_{\Sigma'} = x_1 \xrightarrow{\sigma_1} x_2 \cdots \xrightarrow{\sigma_{q-1}} x_q$ with $\Sigma' \subseteq \Sigma$ is a path in the FSM such that $\sigma_i \in \Sigma'\ \forall i \in \{1, \ldots q-1\}$ (note that the path may consist of a single state only). The following functions are defined for every event path:*

- *$Start(P_{\Sigma'})$ returns the start state of path $P_{\Sigma'}$*
- *$Targ(P_{\Sigma'})$ returns the target state of path $P_{\Sigma'}$*
- *$EvSet(P_{\Sigma'}, \dot{\Sigma})$ returns the events in $\dot{\Sigma} \subseteq \Sigma'$ of path $P_{\Sigma'}$.*

Determining the diagnosis information amounts to looking at every path P in the global model from the initial state x_0 to a state x whose observable event sequence corresponds to the event sequence S actually observed. A tuple (x, l) is part of the diagnosis, if the fault label l corresponds to the faulty event set $EvSet(P, \Sigma_f)$ of path P.

Definition 2 (Diagnosis information) *The diagnosis information $\phi_S \in 2^{X \times 2^{\Sigma_f}}$ that is consistent with a global model $G = \langle X, \Sigma, x_0, T \rangle$ and a sequence of observable events $S = [o_1, \ldots, o_k]$ is defined as follows:*

$$\phi_S = \{(x, l) \mid \overset{1}{P_{\Sigma_u}} \xrightarrow{o_1} \overset{2}{P_{\Sigma_u}} \cdots \xrightarrow{o_{k-1}} \overset{k}{P_{\Sigma_u}} \xrightarrow{o_k} x \text{ is a path in } G \text{ with } l = \bigcup_{j=1}^{k} EvSet(\overset{j}{P_{\Sigma_u}}, \Sigma_f) \text{ and } Start(\overset{1}{P_{\Sigma_u}}) = x_0\}.$$

The aim of on-line diagnosis approaches is to provide timely diagnosis information. This can be done most efficiently following the classical diagnoser approach [8]. A diagnoser is a deterministic finite state machine whose transitions are all labelled with observable events and whose states are directly labelled by the diagnosis information that is consistent with the past observations. Hence it is a means for efficiently retrieving the set of faults that have occured. It does, however, no longer allow inferences about the order in which the faults have occured. Figure 2 depicts a small part of the diagnoser for the system shown in Figure 1. For instance, $\phi_{[CS1obs, SWobs]}$ is found by following the transition sequence $q_0 \xrightarrow{CS1obs} q_1 \xrightarrow{SWobs} q_2$ and by retrieving the state label of q_2. Unfortunately, diagnosers can be so large that they are not computable for all but the smallest applications. In this paper we present a decentralised diagnosis approach that only precomputes the diagnosis information for each component. Since this computation is done locally only, these diagnosis models remain computable even for large applications.

3 The Local Diagnosis Model

The diagnosis model $\mathtt{G}_i$ is a sort of local diagnoser for the component in which the fault behaviours are abstracted and accounted for in the state labels of a FSM. To ensure that this model allows the retrieval

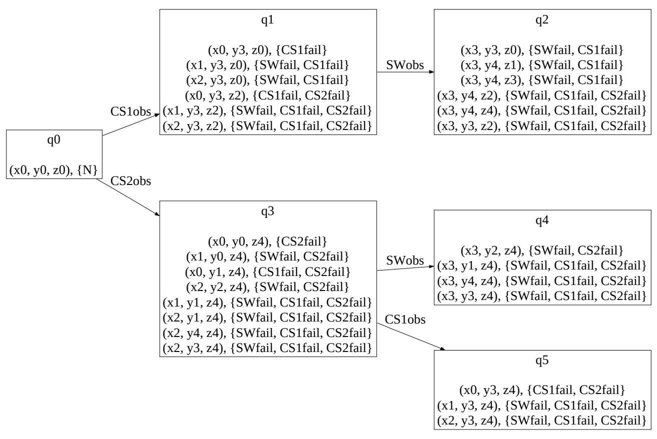

q1
(x0, y3, z0), {CS1fail}
(x1, y3, z0), {SWfail, CS1fail}
(x2, y3, z0), {SWfail, CS1fail}
(x0, y3, z2), {CS1fail, CS2fail}
(x1, y3, z2), {SWfail, CS1fail, CS2fail}
(x2, y3, z2), {SWfail, CS1fail, CS2fail}
SWobs
q2
(x3, y3, z0), {SWfail, CS1fail}
(x3, y4, z1), {SWfail, CS1fail}
(x3, y4, z3), {SWfail, CS1fail}
(x3, y4, z2), {SWfail, CS1fail, CS2fail}
(x3, y4, z4), {SWfail, CS1fail, CS2fail}
(x3, y3, z2), {SWfail, CS1fail, CS2fail}
CS1obs
q0
(x0, y0, z0), {N}
CS2obs
q3
(x0, y0, z4), {CS2fail}
(x1, y0, z4), {SWfail, CS2fail}
(x0, y1, z4), {CS1fail, CS2fail}
(x2, y2, z4), {SWfail, CS2fail}
(x1, y1, z4), {SWfail, CS1fail, CS2fail}
(x2, y1, z4), {SWfail, CS1fail, CS2fail}
(x2, y4, z4), {SWfail, CS1fail, CS2fail}
(x2, y3, z4), {SWfail, CS1fail, CS2fail}
SWobs
q4
(x3, y2, z4), {SWfail, CS2fail}
(x3, y1, z4), {SWfail, CS1fail, CS2fail}
(x3, y4, z4), {SWfail, CS1fail, CS2fail}
(x3, y3, z4), {SWfail, CS1fail, CS2fail}
CS1obs
q5
(x0, y3, z4), {CS1fail, CS2fail}
(x1, y3, z4), {SWfail, CS1fail, CS2fail}
(x2, y3, z4), {SWfail, CS1fail, CS2fail}

Figure 2. (Part of the) diagnoser for the example shown in Figure 1.

of the global diagnosis information on the basis of the local one we label the transitions of $\mathtt{G}_i$ with observable and shared events and use two state labelling functions:

- the usual or "classical" labelling function $\mathtt{R}_i$ which retrieves the diagnosis information that is consistent with any sequence of events S_i observed by component G_i, and
- the *quiet* labelling function $\mathtt{R}'_i$ which retrieves the diagnosis information that is consistent with S_i, followed by any sequence of fault events of G_i.

Hence, to determine which system states and faults are possible immediately after the last observation of S_i, we use the classical function. In contrast, to determine the diagnosis information that is consistent with S_i and any future behaviour that does not involve other components and that cannot be observed, we use the quiet labelling function.

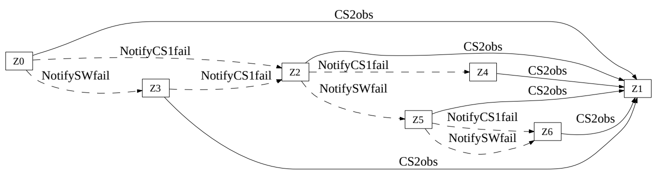
Definition 3 (Local diagnosis model) *The local diagnosis model of a component $G_i = \langle X_i, \Sigma_i, x_{0_i}, T_i \rangle$ is the deterministic finite state machine $\mathtt{G}_i = \langle \mathtt{X}_i, \Psi_i, \Sigma'_i, \mathtt{R}_i, \mathtt{R}'_i, \mathtt{x}_{0_i}, \mathtt{T}_i \rangle$ where $\mathtt{X}_i$ is the set of states, $\Psi_i = X_i \times 2^{\Sigma_{f_i}}$ is the set of possible local diagnosis candidates, $\Sigma'_i = \Sigma_{s_i} \cup \Sigma_{o_i}$ is the set of events, $\mathtt{x}_{0_i}$ is the initial state, $\mathtt{T}_i \subseteq \mathtt{X}_i \times \Sigma'_i \times \mathtt{X}_i$ is the set of transitions, and $\mathtt{R}_i$ and $\mathtt{R}'_i$ are the classical and quiet state labelling function that associate a state to its possible local diagnosis information ($\mathtt{R}_i : \mathtt{X}_i \mapsto 2^{\Psi_i}$ and $\mathtt{R}'_i : \mathtt{X}_i \mapsto 2^{\Psi_i}$). $\mathtt{R}_i$, $\mathtt{R}'_i$ and $\mathtt{T}_i$ satisfy:*
$\mathtt{R}_i(\mathtt{x}_{0_i}) = \{(x_{0_i}, \emptyset)\}$ and

$\mathtt{x}_i \xrightarrow{\sigma} \mathtt{x}'_i \in \mathtt{T}_i$ iff

$$\begin{aligned}\mathtt{R}_i(\mathtt{x}'_i) = \{(x'_i, l_i \cup l'_i) \mid\ & \exists (x_i, l_i) \in \mathtt{R}_i(\mathtt{x}_i) \text{ such that } P_{\Sigma_{f_i}} \xrightarrow{\sigma} x'_i \\ & \text{is a path in } G_i \text{ with } Start(P_{\Sigma_{f_i}}) = x_i \\ & \text{and } l'_i = EvSet(P_{\Sigma_{f_i}}, \Sigma_{f_i})\} \text{ and}\end{aligned}$$

$$\begin{aligned}\mathtt{R}'_i(\mathtt{x}_i) = \mathtt{R}_i(\mathtt{x}_i) \cup \{(x_j, l_i \cup l'_i) \mid\ & P_{\Sigma_{f_i}} \text{ is a path in } G_i \text{ with} \\ & Start(P_{\Sigma_{f_i}}) = x_i,\ Targ(P_{\Sigma_{f_i}}) = x_j, \\ & l'_i = EvSet(P_{\Sigma_{f_i}}, \Sigma_{f_i}),\ (x_i, l_i) \in \mathtt{R}_i(\mathtt{x}_i)\}.\end{aligned}$$

Figure 3 shows the local diagnosis model for $CS2$. The states Z_i on the left hand side of the equalities are those of the local diagnosis model shown in the figure, whilst the states z_i on the right hand



$$\mathtt{R}_i(Z_0) = \Big\{(z_0, \emptyset)\Big\} \qquad \mathtt{R}'_i(Z_0) = \Big\{(z_0, \emptyset),\ (z_2, \{CS2fail\})\Big\}$$
$$\mathtt{R}_i(Z_1) = \mathtt{R}'_i(Z_1) = \Big\{(z_4, \{CS2fail\})\Big\}$$
$$\mathtt{R}_i(Z_2) = \mathtt{R}'_i(Z_2) = \Big\{(z_1, \emptyset),\ (z_2, \{CS2fail\})\Big\}$$
$$\mathtt{R}_i(Z_3) = \mathtt{R}'_i(Z_3) = \Big\{(z_0, \emptyset),\ (z_2, \{CS2fail\})\Big\}$$
$$\mathtt{R}_i(Z_4) = \mathtt{R}'_i(Z_4) = \Big\{(z_2, \{CS2fail\})\Big\}$$
$$\mathtt{R}_i(Z_5) = \mathtt{R}'_i(Z_5) = \Big\{(z_3, \emptyset),\ (z_2, \{CS2fail\}),\ (z_4, \{CS2fail\})\Big\}$$
$$\mathtt{R}_i(Z_6) = \mathtt{R}'_i(Z_6) = \Big\{(z_2, \{CS2fail\}),\ (z_4, \{CS2fail\})\Big\}$$

Figure 3. Local diagnosis model of $CS2$ depicted in Figure 1.

side are those of the component $CS2$ in Figure 1. Note that the number of different states in $\mathtt{G}_i$ only depends on the number of different classical state labels. Two local diagnosis states are identical iff their classical labels are identical (see definition of $\mathtt{T}_i$). In that case, the quiet labels are necessarily identical (see definition of $\mathtt{R}'_i$). However two different states can also have identical quiet labels (see for instance states Z_0 and Z_3 in Figure 3).

4 Computation of Global Diagnosis Information

Computing the global diagnosis information on the basis of local diagnosis models requires both: a synchronisation of the individual transition sets and a synchronisation of the state labels. We now define these synchronisation rules by introducing a new model.

The centralised diagnosis model $\mathtt{G} = \langle \mathtt{X}, \Psi, \Sigma', \mathtt{R}, \mathtt{x}_0, \psi_0, \mathtt{T}\rangle$ is defined as the synchronous composition of the local diagnosis models such that $\mathtt{X} = \prod_{i=1}^{n} \mathtt{X}_i$ is the set of centralised model states, $\Psi = \prod_{i=1}^{n} \Psi_i$ is the set of possible global diagnosis information, $\Sigma' = \Sigma_o \cup \Sigma_s$ are the events ($\Sigma_o = \bigcup_{i=1}^{n} \Sigma_{o_i}$ and $\Sigma_s = \bigcup_{i=1}^{n} \Sigma_{s_i}$), $\mathtt{x}_0 = \prod_{i=1}^{n} \mathtt{x}_{0_i}$ is the initial state, $\psi_0 = \{(x_0, \emptyset)\}$ is the initial global diagnosis information, $\mathtt{T}$ is the transition set, and $\mathtt{R}$ is the diagnosis labelling function ($\mathtt{R} : \Sigma_o \times \mathtt{X} \mapsto 2^{\Psi}$). Let σ denote an observation defined in component G_j and $\mathtt{x} = \prod_i \mathtt{x}_i$ a state in $\mathtt{G}$. Then $\mathtt{R}$ satisfies:

$$\mathtt{R}(\sigma, \mathtt{x}) = \mathtt{R}_j(\mathtt{x}_j) \times \prod_{i, i \neq j} \mathtt{R}'_i(\mathtt{x}_i).$$

The transitions are defined as follows:

$$\begin{aligned} \mathtt{T} = \{(\mathtt{x}_1, \ldots, \mathtt{x}_n) \xrightarrow{\sigma} (\mathtt{x}'_1, \ldots, \mathtt{x}'_n) \mid \\ \forall i \in 1 \ldots n \text{ s.t. } \sigma \in \Sigma'_i, \mathtt{x}_i \xrightarrow{\sigma} \mathtt{x}'_i \in \mathtt{T}_i \text{ and} \\ \forall i \in 1 \ldots n \text{ s.t. } \sigma \notin \Sigma'_i, \mathtt{x}_i = \mathtt{x}'_i\}. \end{aligned}$$

Thus the global diagnosis information is derived as composition of the local diagnosis information. In general, it does not only depend on the centralised state itself but also on the observation leading to that state, or more precisely on the component G_j that observed the last event. Only for this local diagnosis model $\mathtt{G}_j$ the information is obtained using the classical labelling function $\mathtt{R}_j$. The local diagnosis information of the other components is retrieved via the quiet function $\mathtt{R}'_i$ as stated in the following proposition:

Proposition 1 (Diagnosis Retrieval)
Let $\mathtt{G} = \langle \mathtt{X}, \Psi, \Sigma', \mathtt{R}, \mathtt{x}_0, \psi_0, \mathtt{T}\rangle$ denote the global diagnosis model of a system G and $S = [o_1, \ldots, o_k]$ a sequence of observable events. Then

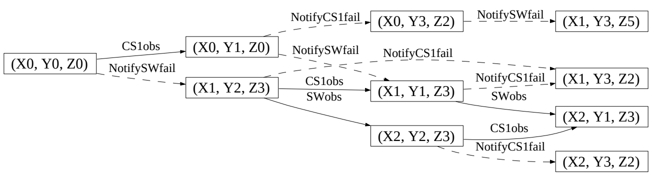


Figure 4. (Part of) centralised diagnosis model for the component models depicted in Figure 1.

$$\begin{aligned} \phi_S = \{\mathtt{R}(o_k, \mathtt{x}) \mid \quad & P = P^1_{\Sigma_s} \xrightarrow{o_1} P^2_{\Sigma_s} \cdots \xrightarrow{o_{k-1}} P^k_{\Sigma_s} \xrightarrow{o_k} \mathtt{x} \\ & \textit{is a path in } \mathtt{G} \textit{ with } Start(P^1_{\Sigma_s}) = \mathtt{x}_0\}. \end{aligned}$$

Note, also the on-line diagnosis approach based on the local diagnosis models makes use of this proposition. It computes all the paths P by triggering transitions in local models only (see Section 7).

Figure 4 shows a part of the centralised diagnosis model for our example application. For instance, considering the sequence $S = [CS1obs]$ there are two paths that satisfy the condition of Proposition 1: $(X_0, Y_0, Z_0) \xrightarrow{CS1obs} (X_0, Y_1, Z_0)$ and $(X_0, Y_0, Z_0) \xrightarrow{NotifySWfail} (X_1, Y_2, Z_3) \xrightarrow{CS1obs} (X_1, Y_1, Z_3)$. Thus, the global diagnosis information that is consistent with S is:

$$\begin{aligned} \phi_S &= \mathtt{R}(CS1obs, (X_0, Y_1, Z_0)) \cup \mathtt{R}(CS1obs, (X_1, Y_1, Z_3)) \\ &= (\mathtt{R}'_{SW}(X_0) \times \mathtt{R}_{CS1}(Y_1) \times \mathtt{R}'_{CS2}(Z_0)) \cup \\ &\quad (\mathtt{R}'_{SW}(X_1) \times \mathtt{R}_{CS1}(Y_1) \times \mathtt{R}'_{CS2}(Z_3)) \\ &= \Big(\Big\{(x_0, \emptyset),\ (x_1, \{SWfail\})\Big\} \times \Big\{(y_3, \{CS1fail\})\Big\} \times \\ &\qquad \Big\{(z_0, \emptyset),\ (z_2, \{CS2fail\})\Big\}\Big) \cup \\ &\quad \Big(\Big\{(x_2, \{SWfail\})\Big\} \times \Big\{(y_3, \{CS1fail\})\Big\} \times \\ &\qquad \Big\{(z_0, \emptyset),\ (z_2, \{CS2fail\})\Big\}\Big) \\ &= \Big\{(x_0, \emptyset),\ (x_1, \{SWfail\}), (x_2, \{SWfail\})\Big\} \times \\ &\qquad \Big\{(y_3, \{CS1fail\})\Big\} \times \Big\{(z_0, \emptyset),\ (z_2, \{CS2fail\})\Big\}. \end{aligned}$$

It corresponds exactly to the label of diagnoser state q_1 shown in Figure 2 and was determined without the expensive and often infeasible diagnoser computation. Moreover, in contrast to the diagnosis computation based on the global model (see Definition 2) it is no longer necessary to determine the fault events of a path and to update this fault set with the arrival of a new observation. Instead we only need to synchronise state sets to retrieve the global diagnosis information based on the local ones. Unlike the update of fault labels, this operation is very efficient when performed symbolically (see Section 7). Thus, we can exploit the advantages of symbolic representations and computations more exhaustively than existing symbolic diagnosis approaches.

5 Symbolic Local Diagnosis Model Representation

Before we describe the encoding of our symbolic models we briefly review binary decision diagrams. Ordered binary decision diagrams (OBDDs, or BDDs for short) [2] are a form of reduced decision graph that provide a compact canonical representation of boolean functions $\mathcal{B}^n \mapsto \mathcal{B}$. While the BDD representation still requires exponential space in the number of boolean variables in the worst case, the reductions often make the BDD of a function much smaller than its disjunctive normal form (DNF). Any boolean operation $f \star g$ on two BDDs f and g, can be carried out in $\mathbb{O}(|f||g|)$ at most, where $|f|$ denotes the number of nodes in the BDD f. BDDs are particularly suitable to represent and manipulate large state and transition sets.

Following the general procedure for encoding states and events of FSMs with BDDs we have introduced at least $Nr(Q) = \lceil \log_2 |Q| \rceil$ boolean variables for each set Q which allows us to write state and event numbers in binary. Furthermore, the symbolic representation of transitions requires the introduction of two sets of state variables in order to distinguish between the start and target state of a transition. Each transition is then given as a conjunction involving the state variables, event variables, and primed, i.e. target state, variables. To encode the shared transitions $\mathtt{T}_{s_i}$ of our local diagnosis models we have introduced the following three variable sets:

- $b_i^{\mathtt{X}} = \{b_{1_i}^{\mathtt{X}}, \ldots, b_{Nr(\mathtt{X}_i)_i}^{\mathtt{X}}\}$ are the state variables used to represent start states of transitions;
- $b_i^{\mathtt{X}'} = \{b_{1_i}^{\mathtt{X}'}, \ldots, b_{Nr(\mathtt{X}_i)_i}^{\mathtt{X}'}\}$ are the primed variables used to represent target states of transitions; and
- $b^S = \{b_1^S, \ldots, b_{Nr(\Sigma_s)}^S\}$ are the shared event variables of the system.

For instance, state Z_2 of the local diagnoser model depicted in Figure 3 is given by the conjunction $\neg b_3^Z \wedge b_2^Z \wedge \neg b_1^Z$, and the set of states $\{Z_2, Z_5\}$ by the DNF $(\neg b_3^Z \wedge b_2^Z \wedge \neg b_1^Z) \vee (b_3^Z \wedge \neg b_2^Z \wedge b_1^Z)$. The transition $t = Z_2 \xrightarrow{NotifySWfail} Z_5$ is given by $t = (\neg b_3^Z \wedge b_2^Z \wedge \neg b_1^Z) \wedge (\neg b_1^S) \wedge (b_3^{Z'} \wedge \neg b_2^{Z'} \wedge b_1^{Z'})$ and the transition relation $\mathtt{T}_{s_i}$ is then encoded as a DNF.

The encoding of the shared events of all models over the same set of variables b^S allows the efficient on-line synchronisation of all local diagnosis models. In contrast, observable events $b_i^O = b^C \cup b_{loc}^O$ are encoded by a set of local variables b_{loc}^O that represent $\mathtt{G}_i$'s observable events, together with an *identifier* of model $\mathtt{G}_i$ encoded by a set of global variables b^C. The local variables are reused across different models and hence the identifier is necessary to make the observable events globally distinguishable. Furthermore the latter allows the direct retrieval of the component that emitted an observable event which is an important information for the on-line diagnosis approach based on the local diagnosis models (see next section).

The FSM of the symbolic local diagnosis model $\mathtt{G}_i = \langle b_i^{\mathtt{X}}, b_i^{\mathtt{X}'}, b_i^X, b_i^O, b_i^F, b^S, \Phi_i, \Phi_i', \mathtt{x}_{0_i}, \mathtt{T}_{o_i}, \mathtt{T}_{s_i} \rangle$ is thus described via the three BDDs $\mathtt{x}_{0_i}$, $\mathtt{T}_{o_i}$ and $\mathtt{T}_{s_i}$ that represent the initial state, and the observable and shared transitions respectively. The state labelling functions of $\mathtt{G}_i$ are represented by the two BDDs Φ_i and Φ_i' that are encoded over the variables $b_i^{\mathtt{X}} \cup b_i^X \cup b_i^F$, where $b_i^X = \{b_{1_i}^X, \ldots, b_{Nr(X_i)_i}^X\}$ and $b_i^F = \{b_{1_i}^F, \ldots, b_{|\Sigma_{f_i}|i}^F\}$ encode the component states and faults of the diagnosis information. Note since multiple faults can occur simultaneously we have introduced one fault variable for each fault event.

For instance, the state labelling functions for Z_2 of the local diagnoser model depicted in Figure 3, i.e. $\mathtt{R}_i(Z_2) = \mathtt{R}_i'(Z_2) = \{(z_1, \emptyset),\ (z_2, \{CS2fail\})\}$, is encoded as:

$$
\begin{array}{ll}
 & ((\neg b_3^{\mathtt{Z}} \wedge b_2^{\mathtt{Z}} \wedge \neg b_1^{\mathtt{Z}}) \wedge (\neg b_3^Z \wedge \neg b_2^Z \wedge b_1^Z) \wedge (\neg b_{CS2fail}^F)) \\
\vee & ((\neg b_3^{\mathtt{Z}} \wedge b_2^{\mathtt{Z}} \wedge \neg b_1^{\mathtt{Z}}) \wedge (\neg b_3^Z \wedge b_2^Z \wedge \neg b_1^Z) \wedge (b_{CS2fail}^F)).
\end{array}
$$

The symbolic representation of our local diagnosis models is now the basis for the efficient on-line diagnosis approach described next.

6 Symbolic On-line Diagnosis

On-line diagnosis aims to detect faults while the system is working. Given a sequence of observations, it identifies all the faults and system states that are consistent with the occurrence of these events. We now show how we can exploit the symbolic representations of our local diagnosis models to efficiently retrieve this diagnosis information $diagInfo$. More precisely, we describe how the set of states $\mathtt{X}_{Diag}$ can be derived from which we can determine the diagnosis information using our two local state labelling functions. Initially it consists of the initial state $\mathtt{x}_0 = \bigwedge_{i \in \{1,\ldots,n\}} \mathtt{x}_{0_i}$. Then, after each new observation, it is updated symbolically using the basic boolean operations and the following ones:

- IsDef(bdd) returns true iff *bdd* does not represent *false*,
- Extract(bdd, B)
  deletes from *bdd* all occurrences of variables not in B,
- Abstract(bdd, B)
  deletes from *bdd* all occurrences of variables in B,
- Swap(bdd, $\{a_1, \ldots, a_k\}, \{b_1, \ldots, b_k\}$)
  renames, in *bdd*, variable a_i with b_i, $i = 1 \ldots k$, and vice versa.

Algorithm 1 $DiagOnline(\mathtt{G}_i,\ \mathtt{X}_{Diag},\ \sigma)$

```
 1: INPUT:   $\mathtt{X}_{Diag}$ : states whose labels contain the current
                      diagnosis information
             $\sigma$ : new observation
Initialise
 2: $comp \leftarrow GetComp(\sigma)$
Trigger shared events
 3: $X_{new} \leftarrow \mathtt{X}_{Diag}$
 4: while there are new states (i.e. as long as $IsDef(X_{new})$) do
 5:     $\mathtt{T}_{new} \leftarrow X_{new} \wedge \bigwedge_{i \in \{1,\ldots,n\}} (\mathtt{T}_{s_i} \vee \mathtt{T}_{steady_i})$
 6:     $\mathtt{X}_{Targ} \leftarrow SwapVar(ExtractVar(\mathtt{T}_{new}, b^{\mathtt{X}'}), b^{\mathtt{X}}, b^{\mathtt{X}'})$
 7:     $\mathtt{X}_{new} \leftarrow \mathtt{X}_{Targ} \wedge \neg \mathtt{X}_{Diag}$
 8:     $\mathtt{X}_{Diag} \leftarrow \mathtt{X}_{Diag} \vee \mathtt{X}_{new}$
Trigger new observation $\sigma$
 9: $\mathtt{X}_{Targ} \leftarrow SwapVar(\sigma \wedge \mathtt{X}_{Diag} \wedge T_{o_{comp}}, b_{comp}^{\mathtt{X}}, b_{comp}^{\mathtt{X}'})$
10: $\mathtt{X}_{Targ} \leftarrow ExtractVar(\mathtt{X}_{Targ}, b_{comp}^{\mathtt{X}})$
11: $\mathtt{X}_{newDiag} \leftarrow \mathtt{X}_{Targ}$
Look up diagnosis information
12: $diagInfo \leftarrow AbstractVar(\mathtt{X}_{Targ} \wedge \Phi_{comp}, b_{comp}^{\mathtt{X}})$
13: for all components $j \neq comp$ do
14:     $diagInfo \leftarrow AbstractVar(diagInfo \wedge \Phi_j', b_j^{\mathtt{X}})$
15: OUTPUT:
    states $\mathtt{X}_{newDiag}$ whose labels contain the new global diagnosis
    information $diagInfo$
```

Algorithm 1 describes how the diagnosis information is updated based on our local diagnosis models following a new observable event σ. In order to retrieve this information, we need to consider the classical label Φ_{comp} of the local diagnosis model $\mathtt{G}_{comp}$ in which σ is defined and the quiet state labels Φ_j' of the remaining models. Hence we need to determine the component in which σ is defined. We do this using function $GetComp$ (line 2) that returns the corresponding identifier of the component by considering the b^C variables used to encode σ (see Section 5).

Now, to update the diagnosis information, we first consider all possible interactions among components that could have taken place after the last event was observed and before the new observation σ (lines 3–8). This requires the triggering of all shared transitions from states in $\mathtt{X}_{Diag}$. In order to perform this operation we have to deal with the fact that different shared events are defined for different sets of components[5]. This gives us two options for triggering shared transitions:

[5] For instance, in our example the event $NotifySWfail$ is defined for all components while the event $NotifyCS1fail$ is only defined for the two control stations $CS1$ and $CS2$.

- We can determine for each shared event σ_s the set of models that define it, i.e. for which $IsDef(\sigma_s \wedge \mathtt{T}_{s_i})$ holds and then trigger the transition in these models only. Hence we would need to trigger shared transition for each event separately.
- For every model $\mathtt{G}_i$ for which σ_s is not defined we can add a *steady* transition $\mathtt{x}_i \xrightarrow{\sigma_s} \mathtt{x}_i$ for all states in $\mathtt{G}_i$. This ensures that all shared events are defined in all models. If such a model is in state $\mathtt{x}_i$ and a shared event σ_s is triggered in another component, the transition $\mathtt{x} \xrightarrow{\sigma_s} \mathtt{x}$ is triggered at the same time. Thus we can trigger all shared transitions at once.

Since the triggering of transition sets is symbolically very efficient, we have chosen the latter option and have added the set of steady transitions $\mathtt{T}_{steady_i}$ to each local diagnosis model. Now we can trigger all shared transitions from all states $\mathtt{X}_{Diag}$ at once (see line 5). In order to determine all sequences of shared events starting in a state in $\mathtt{X}_{Diag}$ we need to repeat the latter operation.

Until a fixed point is reached, all transitions starting in newly retrieved states X_{new} are triggered and the targets states that have not yet been encountered are added to $\mathtt{X}_{Diag}$ (line 8). To obtain the target states $\mathtt{X}_{targ}$ of the transitions $\mathtt{T}_{new}$, we abstract the latter from its start states and events using function $ExtractVar$ (line 6). Originally the targets $\mathtt{X}_{targ}$ are defined over the variables $b^{\mathtt{X}'}$. In order to compute the transitions starting in states of $\mathtt{X}_{targ}$ in the next loop iteration, we swap the state variables to represent X_{targ} over the variables $b^{\mathtt{X}}$ (line 6). Finally, to guarantee the termination of the algorithm, we only consider those targets from which transitions have not yet been triggered. Hence we subtract all previously encountered states $\mathtt{X}_{Diag}$ from X_{targ} (line 7).

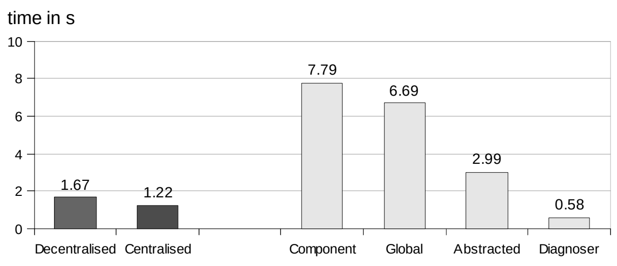
In order to retrieve the states that refer to the diagnosis information that is also consistent with observing σ, we trigger this transition from the previously computed states (lines 9–11). Note that we need to consider only the local observable transition and swap local state variables of the model $\mathtt{G}_{comp}$ in which σ is defined (line 9).

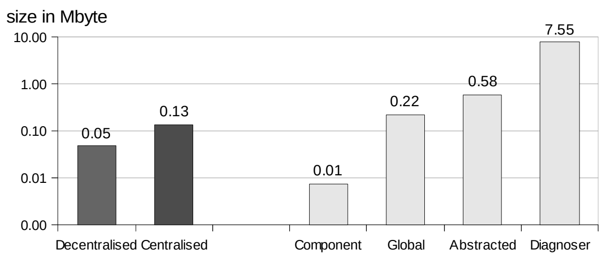
Finally, the diagnosis information is obtained by combining the labels of the states computed above which is done according to the synchronisation rules defined in Section 4. Symbolically this is implemented using the $\wedge$ operator (lines 12–14).

In order to compare the performance of our decentralised diagnosis approach also with a diagnosis method based on the centralised diagnosis model we have implemented such an approach as well. The symbolic centralised diagnosis model $\mathtt{G} = \langle b^{\mathtt{X}}, b^{\mathtt{X}'}, b^{X}, b^{O}, b^{S}, b^{F}, \Phi_1, \ldots, \Phi_n, \Phi'_1, \ldots, \Phi'_n, \mathtt{x}_0, \mathtt{T}_o, \mathtt{T}_s\rangle$ is described by the three additional BDDs $\mathtt{x}_0, \mathtt{T}_o$, and $\mathtt{T}_s$ that are defined over the state variables $b^{\mathtt{X}} = \cup_{i=1}^{n} b_i^{\mathtt{X}}$ and $b^{\mathtt{X}'} = \cup_{i=1}^{n} b_i^{\mathtt{X}'}$. Note that we did not synchronise the state labelling functions since it can be efficiently retrieved on-line (lines 12–14) and thus would only increase the space required for the model representation. The on-line diagnosis based on $\mathtt{G}$ is analogue to the computation described in Algorithm 1. Instead of triggering local transitions in lines 5 and 9 we now need to trigger transitions in the centralised model.

7 Experimental Evaluation

We implemented our approach on top of the CUDD BDD package (`http://vlsi.colorado.edu/~fabio/CUDD`) and run our experiments on a 3.2 GHz Pentium IV with 1 Gbyte of memory. For conducting our evaluation we used a system derived from our example application as a case study [7]. It consists of 3 components (a switch with 12 states and 18 transitions, a primary control station of 13 states and 15 transitions, and a backup control station of 19 states and 28 transitions), 9 observable events, 11 fault types, and 8 other unobservable events. We generated by simulation 100 arbitrary scenarios (possible sequences of observations) of 10,000 observations[6] each, and used them as input to all models.





	Decentralised	Centralised	Component	Global	Abstracted	Diagnoser
State Nr.	Ø 70	3322	Ø 17.67	1063	965	18474
Trans. Nr.	Ø 250	10436	Ø 34	2912	48968	120698

Figure 5. At the top: average diagnosis times for our models (left) and for the symbolic models presented in [10] (right), based on 100 scenarios consisting each of a sequence of 10,000 observations. At the bottom: corresponding model sizes.

We have compared our approach with the closest work to ours namely the symbolic diagnosis methods described in [10]. These were based on the component, global, and diagnoser model which are all introduced in Section 2. The component based diagnosis approach computes first all paths that satisfy the condition stated in Definition 2. From them it then extracts the labels of fault transitions and determines the set of faults that are consistent with the observed events. This diagnosis method can best be compared to the one based on our local diagnosers where we also perform local computations to obtain on-line all paths satisfying a similar condition, namely the one stated in Proposition 1. However, in contrast to the component based approach we do not need to determine the fault events of a path and to update the fault sets. The latter is symbolically very inefficient since it requires the isolation and manipulation of individual boolean variables. In our approach this was done off-line. The diagnosis approach using the global model is comparable to our method based on the centralised diagnosis model (see Section 4 and the end of Section 6). Both approaches can resort to a synchronised representation of the system for computing on-line all paths that are consistent with Definition 2 and Proposition 1 respectively. The retrieval of the fault information for the global model based approach is the same as for the component based approach described above and hence very inefficient.

Figure 5 shows how our results compare to the above existing methods. Here we see that apart from the diagnoser approach our diagnosis methods are much faster. This results from the fact that our approach avoids the costly update of fault labels which requires the isolation and manipulation of individual boolean variables.

[6] Unlike the components in Figure 1 which have final states without outgoing transitions, the components of the case study have observable loops (and so does the diagnoser) which is why it was possible to track sequences of 10,000 observable events.

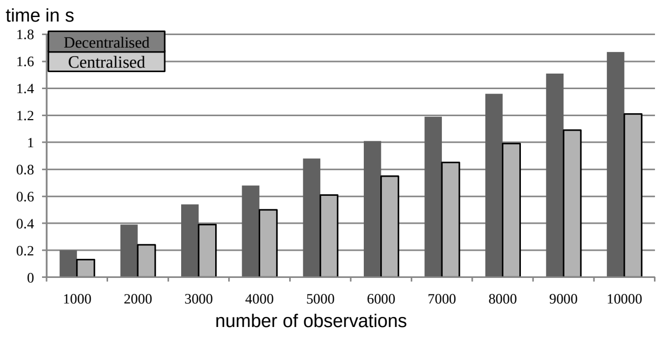


Figure 6. Development of the diagnosis times for our decentralised and centralised diagnosis approaches.

Furthermore the diagnosis times show that our decentralised diagnosis approach is not much slower than the one based on the centralised diagnosis model. This might seem surprising since the online synchronisation implies that much more transition sets need to be triggered in the local diagnosis models than in the centralised one. It can be explained by the fact that the efficiency of triggering transition sets depends on the size of the BDD that represents the transitions. Since our local diagnosis models are smaller they therefore allow for a faster triggering of transition sets.

To consider the time/space complexity tradeoff we now also look at the space requirements for representing the diagnosis models. These were determined by counting the nodes of their BDDs and by multiplying this number with the space requirements to represent one BDD node. The bottom of Figure 5 depicts these results on a logarithmic scale. They show that the small diagnosis time of the diagnoser based approach was achieved at the expense of high space requirements that were more than two orders of magnitude higher than that of our decentralised approach. In contrast, our diagnosis approach took only three times longer.

The results also show that compared to the existing approaches based on the global model and the abstracted one our diagnosis approaches are not only faster but require also less space. In order to see whether our approaches can also be used for updating a continuous flow of observations we need to ensure that our diagnosis times increase only linearly in the number of observations. As Figure 6 shows, this is the case for both of our methods.

8 Conclusion, Related & Future Work

We have presented a decentralised symbolic diagnosis approach that is efficient in terms of both time and space. Run time efficiency was achieved by exploiting the advantages of BDD based representations and computations. The slow computation of the local diagnosis information was performed off-line while the fast BDD operations of synchronising state and transition sets were performed online. Space efficiency, was obtained by performing all computations (whether off-line or on-line) based on local models only. Compared to the spectrum of symbolic approaches presented in [10], which is the closest work to ours, only the symbolic diagnoser approach was faster. However, our diagnosis time had only tripled while the space requirements were reduced by more than two orders of magnitude. Additionally we have shown that our diagnosis time can be further reduced by synchronising the local models off-line. This is at the expense of a larger model representation.

Other work on symbolic discrete-event system diagnosis include that of Micalizio [5]. It describes a simulation-based BDD method for the recovery in which all diagnostic reasoning is performed online. Therefore it is rather comparable to the component based approach described in [10]. In [11] the authors show how BDDs can help to diagnose distributed discrete-event systems modelled as Petri nets. In their approach, which is limited to two components whose interactions can be observed, BDDs are used to encode the diagnosis information. However, the model itself is represented in an nonsymbolic way and the computation of diagnosis information is also nonsymbolic. Hence a continual conversion from nonsymbolic to symbolic representation is required.

Among the nonsymbolic decentralised on-line diagnosis methods the work introduced in [3] is the closest to ours. This approach consists of a set of diagnosers, that each explain the observations from a site, i.e. from a set of components. The states of these diagnosers are labelled by sets of global states and fault labels and the global diagnosis information is obtained based on these state labels and the unobservable reach. The latter is similar to our quiet state labels but can not be determined based on local models only. Since this approach does require the computation of the global model it is unlikely to scale to large systems.

The perspectives of this paper are numerous. We now plan to extend our decentralised diagnosis approach also to subsystems, i.e. instead of computing a local diagnoser for each component we plan to compute one for sets of components. Then we would like to work on methods for optimally decomposing the overall system into subsystems for which the diagnosers are to be computed. Such a decision might not only be based on the available computational resources but also on the number of observations, faults, components and their interactions, since these parameters might impact the efficiency of the diagnosis approach and thus the computational resources required to perform it. Moreover, it would be interesting to combine our approach with symbolic planning and testing techniques in order to develop efficient methods for the autonomous repair of systems.

REFERENCES

[1] P. Baroni, G. Lamperti, P. Pogliano, and M. Zanella, 'Diagnosis of large active systems', *Artificial Intelligence*, **110**(1), 135–183, (May 1999).

[2] R. E. Bryant, 'Graph-based algorithms for boolean function manipulation', *IEEE Transactions on Computers*, **C-35**(8), 677–691, (1986).

[3] R. Debouk, S. Lafortune, and D. Teneketzis, 'Coordinated decentralized protocols for failure diagnosis of discrete event systems', *Journal of Discrete Event Dynamical Systems: Theory and Application*, **10**(1–2), 33–86, (January 2000).

[4] G. Lamperti and M. Zanella, 'Flexible diagnosis of discrete-event systems by similarity-based reasoning techniques', *Artificial Intelligence*, **170**(3), 232–297, (2006).

[5] Roberto Micalizio, 'A distributed control loop for autonomous recovery in a multi-agent plan', in *IJCAI*, pp. 1760–1765, (2009).

[6] Y. Pencolé and M.-O. Cordier, 'A formal framework for the decentralised diagnosis of large scale discrete event systems and its application to telecommunication networks', *Artificial Intelligence*, **164**, 121–170, (May 2005).

[7] L. Rozé and M. O. Cordier, 'Diagnosing discrete-event systems : extending the "diagnoser approach" to deal with telecommunication networks', *Journal on Discrete-Event Dynamic Systems : Theory and Applications*, **12**(1), 43–81, (January 2002).

[8] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis, 'Failure diagnosis using discrete event models', *IEEE Transactions on Control Systems Technology*, **4**(2), 105–124, (1996).

[9] A. Schumann, Y. Pencolé, and S. Thiébaux, 'Diagnosis of discrete-event systems using binary decision diagrams', in *15th International Workshop on Principles of Diagnosis (DX-04)*, pp. 197–202, (2004).

[10] A. Schumann, Y. Pencolé, and S. Thiébaux, 'A spectrum of symbolic on-line diagnosis approaches', in *22nd American National Conference on Artificial Intelligence (AAAI)*, pp. 335–340, (2007).

[11] F. Xue, L. Yan, and D. Zheng, 'Fault diagnosis of distributed discrete event systems using obdd', *Informatica*, **16(3)**, 431–448, (2005).