Sensor Minimization Problems with Static or Dynamic Observers for Fault Diagnosis

Franck Cassez, Stavros Tripakis, Karine Altisen

Sensor Minimization Problems with Static or Dynamic Observers for Fault Diagnosis* (Extended Abstract)

Franck Cassez† Stavros Tripakis‡ Karine Altisen§

Abstract

We study sensor minimization problems in the context of fault diagnosis. Fault diagnosis consists of synthesizing a diagnoser that observes a given plant and identifies faults in the plant as soon as possible after their occurrence. Existing literature on this problem has considered the case of static observers, where the set of observable events does not change during execution of the system. In this paper, we consider static as well as dynamic observers, where the observer can switch sensors on or off, thus dynamically changing the set of events it wishes to observe.

1 Introduction

Monitoring, Testing, Fault Diagnosis and Control. Many problems concerning the monitoring, testing, fault diagnosis and control of discrete event systems (DES) can be formalized by using finite automata over a set of *observable* events Σ, plus a set of *unobservable* events [8, 10]. The invisible actions can often be represented by a single unobservable event ε. Given a finite automaton over $\Sigma \cup \{\varepsilon\}$ which is a model of a *plant* (to be monitored, tested, diagnosed or controlled) and an *objective* (good behaviours, what to test for, faulty behaviours, control objective) we want to check if a monitor/tester/diagnoser/controller exists that achieves the objective, and if possible to synthesize one automatically.

The usual assumption in this setting is that the set of observable events is fixed (and this in turn determines the set of unobservable events as well). Observing an event usually requires some detection mechanism, i.e. , a *sensor* of some sort. Which sensors to use, how many of them, and where to place them are some of the design questions that are often difficult to answer, especially without knowing what these sensors are to be used for.

In this paper we study problems of *sensor minimization*. These problems are interesting since observing an event can be costly in terms of time or energy: computation time must be spent to read and process the information provided by the sensor, and power is required to operate the sensor (as well as to perform the computations). It is then essential that the sensors used really provide useful information. It is also important for the computer to discard any information given by a sensor that is not really needed. In the case of a fixed set of observable events, it is not the case that all sensors always provide useful information and sometimes energy (sensor operation and computer treatment) is spent for nothing. For example, to diagnose a fault in the system described by the automaton $\mathcal{B}$, Figure 3, an observer only has to watch event a, and *when a has occurred*, to watch event b: if the sequence $a.b$ occurs, for sure a fault has occurred and the observer can raise an alarm. It is then not useful to switch on sensor b before an a has occurred.

Sensor Minimization and Fault Diagnosis. We focus our attention on sensor minimization, without looking at problems related to sensor placement, choosing between different types of sensors, and so on. We also focus on a particular observation problem, that of *fault diagnosis*. We believe, however, that the results we obtain are applicable to other contexts as well.

Fault diagnosis consists in observing a plant and detecting whether a fault has occurred or not. We follow the discrete-event system (DES) setting of [9] where the behavior of the plant is known and a model of it is available as a finite-state automaton over $\Sigma \cup \{\varepsilon, f\}$ where Σ is the set of observable events, ε represents the unobservable events, and f is a special unobservable event that corresponds to the faults. Checking *diagnosability* (whether a fault can be detected) for a given plant and a *fixed* set of observable events

*Work supported by the French government project ACI-CORTOS `http://www.lsv.ens-cachan.fr/aci-cortos`

†CNRS/IRCCyN, 1 rue de la Noë, BP 92101, 44321 Nantes Cedex 3, France. Email: franck.cassez@cnrs.irccyn.fr

‡Cadence Berkeley Labs, 1995 University Avenue, Berkeley, CA, 94704, USA, and CNRS, Verimag Laboratory, Centre Equation, 2, avenue de Vignate, 38610 Gières, France. Email: tripakis@cadence.com

§INPG and Verimag Laboratory, Centre Equation, 2, avenue de Vignate, 38610 Gières, France.

can be done in polynomial time [9, 11, 6]. (Notice that synthesizing a diagnoser involves determinization in general, thus cannot be done in polynomial time.)

We examine sensor optimization problems with both *static* and *dynamic* observers. A static observer always observes the same set of events, whereas a dynamic observer can modify the set of events it wishes to observe during the course of the plant execution (this could be implemented by switching sensors on and off in order to save energy, for example).

In the static observer case, we consider both the standard setting of observable/unobservable events as well as the setting where the observer is defined as a *mask* which allows some events to be observable but not *distinguishable* (e.g., see [3]). Our first contribution is to show that the problems of *minimizing* the number of observable events (or distinct observable outcomes in case of the mask) are NP-complete. Membership in NP can be easily derived by reducing these problems to the standard diagnosability problem, once a candidate minimal solution is chosen non-deterministically. NP-hardness can be shown using reductions of well-known NP-hard problems, namely, clique and coloring problems in graphs.

In the dynamic observer case, we assume that an observer can decide after each new observation the set of events it is going to watch. As a second contribution, we provide a definition of the *dynamic observer synthesis problem* and then show that computing a *dynamic observer* for a given plant, can be reduced to a *game problem*.

Related work. NP-hardness of finding minimum-cardinality sets of observable events so that diagnosability holds under the standard, projection-based setting has been previously reported in [11]. Our result of section 3 can be viewed as an alternative shorter proof of this result. Masks have not been considered in [11]. As we show in section 4, a reduction from the mask version of the problem to the standard version is not straightforward. Thus the result in section 4 is useful and new.

The complexity of finding "optimal" observation masks, i.e. a set that cannot be reduced, has been considered in [7] where it was shown that the problem is NP-hard for general properties. [7] also shows that finding optimal observation masks is polynomial for "mask-monotonic" properties where increasing the set of observable (or distinguishable) events preserves the property in question. Diagnosability is a mask-monotonic property. Notice that optimal observation masks are not the same as minimum-cardinality masks that we consider in our work.

In [4], the authors investigate the problem of computing a minimal-cost strategy that allows to find a subset of the set of observable events s.t. the system is diagnosable. It is assumed that each such subset has a known associated cost, as well as a known a-priori probability for achieving diagnosability.

To our knowledge, the problems of synthesizing dynamic observers for diagnosability, studied in Section 5, have not been addressed previously in the literature. The material of this paper is taken from [1] which contains a follow-up part of the present paper that studies the problem of *optimal-cost dynamic observers synthesis* [2].

Organisation of the paper. In Section 2 we fix notation and introduce finite automata with faults to model DES. In Section 3 we show NP-completeness of the sensor minimization problem for the standard projection-based observation setting. In Section 4 we show NP-completeness of the sensor minimization problem for the mask-based setting. In Section 5 we introduce and study dynamic observers. We define dynamic observers and show that the most permissive dynamic observer can be computed as the strategy in a safety 2-player game.

A full version of the paper containing the omitted proofs is available from the web page of the authors.

2. Preliminaries

2.1. Words and Languages

Let Σ be a finite alphabet and $\Sigma^\varepsilon = \Sigma \cup \{\varepsilon\}$. Σ^* is the set of finite words over Σ and contains ε which is also the empty word. A *language* L is any subset of Σ^*. $\Sigma^+ = \Sigma^* \setminus \{\varepsilon\}$. Given two words ρ, ρ' we denote $\rho.\rho'$ the concatenation of ρ and ρ' (which is defined in the usual way). $|\rho|$ stands for the length of the word ρ and $|\rho|_\lambda$ with $\lambda \in \Sigma$ stands for the number of occurrences of λ in ρ. Given $\Sigma_1 \subseteq \Sigma$, we define the *projection* $\boldsymbol{\pi}_{/\Sigma_1} : \Sigma^* \to \Sigma_1^*$ by: $\boldsymbol{\pi}_{/\Sigma_1}(\varepsilon) = \varepsilon$ and for $a \in \Sigma, \rho \in \Sigma^*$, $\boldsymbol{\pi}_{/\Sigma_1}(a.\rho) = a.\boldsymbol{\pi}_{/\Sigma_1}(\rho)$ if $a \in \Sigma_1$ and $\boldsymbol{\pi}_{/\Sigma_1}(\rho)$ otherwise.

2.2. Finite Automata

Let $f \notin \Sigma^\varepsilon$ be a fresh letter that corresponds to the fault action. An *automaton* A is a tuple[1] $(Q, q_0, \Sigma^{\varepsilon,f}, \to)$ with Q a set of states, $q_0 \in Q$ is the initial state, $\to \subseteq Q \times \Sigma^{\varepsilon,f} \times Q$ is the transition relation. If Q is finite, A is a *finite automaton*. We write $q \xrightarrow{\lambda} q'$ if $(q, \lambda, q') \in \to$. For $q \in Q$, $\mathit{en}(q)$ is the set of actions enabled at q. A *run* ρ from state s in A is a sequence of transitions $s_0 \xrightarrow{\lambda_1} s_1 \xrightarrow{\lambda_2} s_2 \cdots s_{n-1} \xrightarrow{\lambda_n} s_n$ s.t. $\lambda_i \in \Sigma^{\varepsilon,f}$ and $s_0 = s$. We let $\mathit{tgt}(\rho) = s_n$. The set of runs from s in A is denoted $\mathit{Runs}(s, A)$ and we define $\mathit{Runs}(A) = \mathit{Runs}(q_0, A)$. The *trace* of the run ρ, denoted $\mathit{tr}(\rho)$, is the word obtained by concatenating the

[1] In this paper we only use finite automata that generate prefix-closed languages, hence we do not need to use a set of final or accepting states.

symbols λ_i appearing in ρ, for those λ_i different from ε. Given a set $R \subseteq \mathit{Runs}(A)$, $\mathit{Tr}(R) = \{\mathit{tr}(\rho)\ \mathit{for}\ \rho \in R\}$ is the set of traces of the runs in R. A run ρ is *k-faulty* if there is some $1 \leq i \leq n$ s.t. $\lambda_i = f$ and $n - i \geq k$. $\mathit{Faulty}_{\geq k}(A)$ is the set of k-faulty runs of A. A run is *faulty* if it is k-faulty for some $k \in \mathbb{N}$ and $\mathit{Faulty}(A)$ denotes the set of faulty runs. It follows that $\mathit{Faulty}_{\geq k+1}(A) \subseteq \mathit{Faulty}_{\geq k}(A) \subseteq \cdots \subseteq \mathit{Faulty}_{\geq 0}(A) = \mathit{Faulty}(A)$. Finally $\mathit{NonFaulty}(A) = \mathit{Runs}(A) \setminus \mathit{Faulty}(A)$ is the set on *non-faulty* runs of A. We let $\mathit{Faulty}^{\mathit{tr}}_{\geq k}(A) = \mathit{Tr}(\mathit{Faulty}_{\geq k}(A))$ and $\mathit{NonFaulty}^{\mathit{tr}}(A) = \mathit{Tr}(\mathit{NonFaulty}(A))$ be the sets of traces of faulty and non-faulty runs.

A word w is *accepted* by A if $w = \mathit{tr}(\rho)$ for some $\rho \in \mathit{Runs}(A)$. The *language* $\mathcal{L}(A)$ of A is the set of words accepted by A. We assume that each faulty run of A of length n can be extended into a run of length $n+1$. This is required for technical reasons (in order to guarantee that the set of faulty runs where sufficient time has elapsed after the fault is well-defined) and can be achieved by adding ε loop transitions to each deadlock state of A. Notice that this transformation does not change the observations produced by the plant, thus, any observer synthesized for the transformed plant also applies to the original one.

Finally *Product of Automata* without ε-transitions are defined in the usual way: they synchronize on common letters.

3. Sensor Minimization with Static Observers

In this section we address the sensor minimization problem for *static observers*. We point out that the result in this section was already obtained in [11] and we only give here an alternative shorter proof. We are given a finite automaton $A = (Q, q_0, \Sigma^{\varepsilon,f}, \rightarrow)$. The maximal set of observable events is Σ (ε is not observable). We want to decide whether there is a subset $\Sigma_o \subsetneq \Sigma$ such that the faults can be detected by observing only events in Σ_o. Moreover, we would like to find an "optimal" such Σ_o.

A *diagnoser* is a device that observes the plant and raises an "alarm" whenever it detects a fault. We allow the diagnoser to raise an alarm not necessarily immediately after the fault occurs, but possibly some time later, as long as this time is bounded by some $k \in \mathbb{N}$. We model time by counting the "moves" the plant makes (including observable and unobservable ones). If the system generates a word ρ but only a subset $\Sigma_o \subseteq \Sigma$ is observable, the diagnoser can only see $\boldsymbol{\pi}_{/\Sigma_o}(\rho)$.

Definition 1 ((Σ_o, k)-Diagnoser) *Let A be a finite automaton over $\Sigma^{\varepsilon,f}$, $k \in \mathbb{N}$, $\Sigma_o \subseteq \Sigma$. A mapping $D : \Sigma_o^* \rightarrow \{0,1\}$ is a* (Σ_o, k)-diagnoser for A *if (i) for each $\rho \in \mathit{NonFaulty}(A)$, $D(\boldsymbol{\pi}_{/\Sigma_o}(\mathit{tr}(\rho))) = 0$, and (ii) for each $\rho \in \mathit{Faulty}_{\geq k}(A)$, $D(\boldsymbol{\pi}_{/\Sigma_o}(\mathit{tr}(\rho))) = 1$.* ■

A is (Σ_o, k)-diagnosable if there is a (Σ_o, k)-diagnoser for A. A is Σ_o-diagnosable if there is some $k \in \mathbb{N}$ s.t. A is (Σ_o, k)-diagnosable.

Example 1 *Let $\mathcal{A}$ be the automaton shown on Figure 1. The run f is in $\mathit{Faulty}_{\geq 0}(\mathcal{A})$, the run $f.a$ is in $\mathit{Faulty}_{\geq 1}(\mathcal{A})$ and $a.\varepsilon^2$ is in $\mathit{NonFaulty}(\mathcal{A})$.*

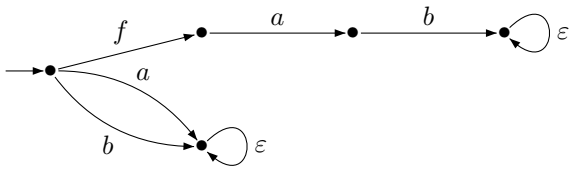


Figure 1. The automaton $\mathcal{A}$

$\mathcal{A}$ is neither $\{a\}$-diagnosable, nor $\{b\}$-diagnosable. This is because, for any k, the faulty run $f.a.b.\varepsilon^k$ gives the same observation as the non-faulty run $a.\varepsilon^k$ (in case a is the observable event) or the non-faulty run $b.\varepsilon^k$ (in case b is the observable event). Consequently, the diagnoser cannot distinguish between the two no matter how long it waits. If both a and b are observable, however, then we can define: $D(a.b.\rho) = 1$ for any $\rho \in \{a,b\}^*$ and $D(\rho) = 0$ otherwise. D is a $(\{a,b\}, 2)$-diagnoser for $\mathcal{A}$.

For given A and Σ_o it is known how to check diagnosability and build a diagnoser (e.g., see [9]). Checking whether A is Σ_o-diagnosable can be done in polynomial time in the size of A, more precisely in $O(|A|^2)$. Computing the minimum k s.t. A is (Σ_o, k)-diagnosable can be done in $O(|A|^3)$. Moreover in case A is Σ_o-diagnosable, there is a diagnoser D that can be represented by a finite automaton. Computing this finite automaton is in $O(2^{|A|})$. Algorithms for solving these problems are given in appendix A in [1] and use the fact that A is (Σ_o, k)-diagnosable iff

$$\boldsymbol{\pi}_{/\Sigma_o}(\mathit{Faulty}^{\mathit{tr}}_{\geq k}(A)) \cap \boldsymbol{\pi}_{/\Sigma_o}(\mathit{NonFaulty}^{\mathit{tr}}(A)) = \emptyset \quad (1)$$

or in other words, there is no pair of runs (ρ_1, ρ_2) with $\rho_1 \in \mathit{Faulty}_{\geq k}(A)$, $\rho_2 \in \mathit{NonFaulty}(A)$ s.t. ρ_1 and ρ_2 give the same observations on Σ_o. In this section we address the problem of *finding* a set of observable events Σ_o that allows faults to be detected. We would like to detect faults using as few observable events as possible.

Problem 1 (Minimum Number of Observable Events)
INPUT: *A, $n \in \mathbb{N}$ s.t. $n \leq |\Sigma|$.*
PROBLEM:

(A) Is there any $\Sigma_o \subseteq \Sigma$ with $|\Sigma_o| = n$, such that A is Σ_o-diagnosable?

(B) If the answer to (A) is "yes", find the minimum n_0 such that there exists $\Sigma_o \subseteq \Sigma$ with $|\Sigma_o| = n_0$ and A is Σ_o-diagnosable.

If we know how to solve Problem 1(A) efficiently then we can also solve Problem 1(B) efficiently: we perform a binary search over n between 0 and $|\Sigma|$, and solve Problem 1(A) for each such n, until we find the minimum n_0 for which Problem 1(A) gives a positive answer.[2] Unfortunately, Problem 1(A) is a combinatorial problem, exponential in $|\Sigma|$, as we show next.

Theorem 1 *Problem 1(A) is NP-complete.*

Proof: (Sketch) Membership in NP is easy: guess a solution Σ_o and check Σ_o-diagnosability (can be done in polynomial time). The proof of NP-hardness is a reduction of the n-clique problem to Problem 1(A). Details can be found in the extended version of the paper. ■

4. Sensor Minimization with Masks

So far we have assumed that observable events are also *distinguishable*. However, there are cases where two events a and b are observable but not distinguishable, that is, the diagnoser knows that a or b occurred, but not which of the two. This is not the same as considering a and b to be unobservable, since in that case the diagnoser would not be able to detect occurrence of a or b. Distinguishability of events is captured by the notion of a *mask*.

Definition 2 (Mask) *A* mask (M, n) *over* Σ *is a total, surjective function* $M : \Sigma \to \{1, \cdots, n\} \cup \{\varepsilon\}$. ■

M induces a morphism $M^* : \Sigma^* \to \{1, \cdots, n\}^*$. For example, if $\Sigma = \{a, b, c, d\}$, $n = 2$ and $M(a) = M(b) = 1$, $M(c) = 2$, $M(d) = \varepsilon$, then we have $M^*(a.b.c.b.d) = 1.1.2.1 = M^*(a.a.d.c.a)$.

Definition 3 ($((M, n), k)$-diagnoser) *Let* (M, n) *be a mask over* Σ. *A mapping* $D : \{1, \cdots, n\}^* \to \{0, 1\}$ *is a* $((M, n), k)$-diagnoser *for* A *if (i) for each* $\rho \in$ *NonFaulty*(A), $D(M^*(\boldsymbol{\pi}_{/\Sigma}(tr(\rho)))) = 0$ *and (ii) for each* $\rho \in$ *Faulty*$_{\geq k}(A)$, $D(M^*(\boldsymbol{\pi}_{/\Sigma}(tr(\rho)))) = 1$. ■

A is $((M, n), k)$-diagnosable if there is a $((M, n), k)$-diagnoser for A. A is (M, n)-diagnosable if there is some k such that A is $((M, n), k)$-diagnosable.

Given A and a mask (M, n), checking whether A is (M, n)-diagnosable can be done in polynomial time. In fact it can be reduced to checking Σ_o-diagnosability of a modified automaton A_M, with $\Sigma_o = \{1, ..., n\}$. A_M is obtained from A by renaming the actions $a \in \Sigma$ by $M(a)$. It can be seen that A is $((M, n)$-diagnosable iff A_M is $\{1, \cdots, n\}$-diagnosable. Notice that A is $((M, n), k)$-diagnosable iff $M^*(\boldsymbol{\pi}_{/\Sigma}(\textit{Faulty}^{tr}_{\geq k}(A))) \cap M^*(\boldsymbol{\pi}_{/\Sigma}(\textit{NonFaulty}^{tr}(A))) = \emptyset$.

As in the previous section, we are mostly interested in minimizing the observability requirements while maintaining diagnosability. In the context of diagnosis with masks, this means minimizing the number n of distinct outputs of the mask M. We thus define the following problem:

Problem 2 (Minimum Mask)
INPUT: A, $n \in \mathbb{N}$ *s.t.* $n \leq |\Sigma|$.
PROBLEM:

(A) Is there any mask (M, n) *such that* A *is* (M, n)*-diagnosable?*

(B) If the answer to (A) is "yes", find the minimum n_0 *such that there is a mask* (M, n_0) *such that* A *is* (M, n_0)*-diagnosable.*

As with Problem 1, if we know how to solve Problem 2(A) efficiently we also know how to solve Problem 2(B) efficiently: again, a binary search on n suffices.

We will prove that Problem 2 is NP-complete. One might think that this result follows easily from Theorem 1. However, this is not the case. Obviously, a solution to Problem 1 provides a solution to Problem 2: assume there exists Σ_o such that A is (Σ_o, k)-diagnosable and $\Sigma_o = \{a_1, ..., a_n\}$; define a mask $M : \Sigma \to \{1, \cdots, n\}$ such that $M(a_i) = i$ and for any $a \in \Sigma \setminus \Sigma_o$, $M(a) = \varepsilon$. Then, A is $((M, n), k)$-diagnosable. However, a positive answer to Problem 2(A) does not necessarily imply a positive answer to Problem 1(A), as shown by the example that follows.

Example 2 *Consider again the automaton* $\mathcal{A}$ *of Figure 1. Let* $M(a) = M(b) = 1$. *Then* $\mathcal{A}$ *is* $((M, 1), 2)$*-diagnosable because we can build a diagnoser* D *defined by:* $D(\varepsilon) = 0, D(1) = 0, D(1^2.\rho) = 1$ *for any* $\rho \in 1^*$. *However, as we said before, there is no strict subset of* $\{a, b\}$ *that allows* $\mathcal{A}$ *to be diagnosed.*

Theorem 2 *Problem 2 is NP-complete.*

Proof: Membership in NP is again justified by the fact that checking whether a guessed mask works can be done in polynomial time (it suffices to rename the events of the system according to M and apply the algorithm of appendix A in [1]). We show NP-hardness using a reduction of the n-coloring problem. The n-coloring problem asks the following: given an undirected graph $G = (V, E)$, is it possible to color the vertices with colors in $\{1, 2, \cdots, n\}$ so that no two adjacent vertices have the same color? Let $G = (V, E)$ be an undirected graph. Let $E = \{e_1, e_2, \cdots, e_j\}$ be the set of edges with $e_i = (u_i, v_i)$. We let $\Sigma = V$ and define the automaton A_G as pictured in Figure 2. The initial state of A_G is q_0. We claim that G is n-colorizable iff A_G is (M, n)-diagnosable.

[2] Notice that knowing n_0 does not imply we know the required set of observable events Σ_o! We can find (one of the possibly many) Σ_o by searching over all possible subsets Σ_o of Σ of size n_0 (there are $C(|\Sigma|, n_0)$ such combinations) and check for each such Σ_o whether A is Σ_o-diagnosable, using the methods described in the appendix A of [1].

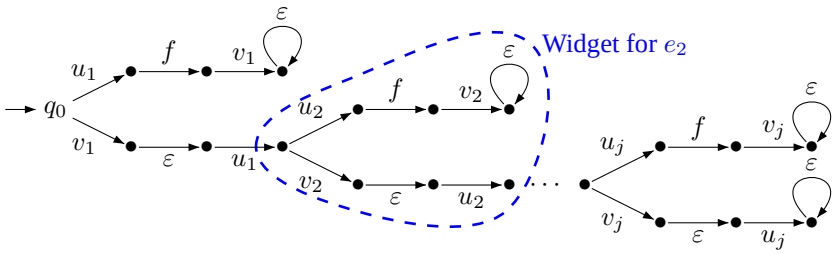


Figure 2. Automaton A_G for n-colorizability

If part. Assume A_G is (M,n)-diagnosable for $n \geq 0$. We first show that for all $i = 1,...,j$, $M(u_i) \neq \varepsilon$, $M(v_i) \neq \varepsilon$ and $M(u_i) \neq M(v_i)$. For any k, we can define the runs $\rho = v_1.\varepsilon.u_1.v_2.\varepsilon.u_2 \cdots u_i.f.v_i.\varepsilon^k$ and $\rho' = v_1.\varepsilon.u_1.v_2.\varepsilon.u_2 \cdots v_i.\varepsilon.u_i$. If either $M(u_i) = \varepsilon$ or $M(v_i) = \varepsilon$ or $M(u_i) = M(v_i)$ holds, then we have $M^*(\boldsymbol{\pi}_{/\Sigma}(tr(\rho))) = M^*(\boldsymbol{\pi}_{/\Sigma}(tr(\rho')))$. This way for any k, there is a faulty run with more than k events after the fault, and a non-faulty run which gives the same observation through the mask. Hence A cannot be $((M,n),k)$-diagnosable for any k and thus A is not (M,n)-diagnosable which contradicts diagnosability of A.

Note that the above implies in particular that $n \geq 1$. We can now prove that G is n-colorizable. Let C be the color mapping defined by $C(v) = M(v)$. We need to prove that $C(u_i) \neq C(v_i)$ for any $(u_i, v_i) \in E$. This holds by construction of A_G and the fact that $M(u_i) \neq M(v_i)$ as shown above.

Only if part. Assume G is n-colorizable. There exists a color mapping $C : V \rightarrow \{1, 2, \cdots, n\}$ s.t. if $(v, v') \in E$ then $C(v) \neq C(v')$. Define the mask M by $M(a) = C(a)$ for $a \in V$. We claim that A_G is $((M,n),1)$-diagnosable (thus, also (M,n)-diagnosable). Assume on the contrary that A_G is not $((M,n),1)$-diagnosable. Then there exist two runs $\rho \in \mathit{Faulty}_{\geq 1}(A_G)$ and $\rho' \in \mathit{NonFaulty}(A_G)$ such that $M^*(\boldsymbol{\pi}_{/\Sigma}(tr(\rho))) = M^*(\boldsymbol{\pi}_{/\Sigma}(tr(\rho')))$. As ρ is 1-faulty it must be of the form $\rho = v_1.\varepsilon.u_1 \cdots u_i.f.v_i.\varepsilon^k$ with $1 \leq i \leq j$ and $k \geq 0$. Notice that $M(a) \neq \varepsilon$ for all $a \in V$. Hence it implies $M^*(\boldsymbol{\pi}_{/\Sigma}(tr(\rho))) = M(v_1).M(u_1) \cdots M(u_i).M(v_i)$, and $|M^*(\boldsymbol{\pi}_{/\Sigma}(tr(\rho)))| = 2i$. Consequently, $|M^*(\boldsymbol{\pi}_{/\Sigma}(\rho'))| = 2i$. The only possible such ρ' which is non-faulty is $\rho' = v_1.\varepsilon.u_1 \cdots v_i.\varepsilon.u_i$. Now, $M^*(\boldsymbol{\pi}_{/\Sigma}(tr(\rho))) = M^*(\boldsymbol{\pi}_{/\Sigma}(tr(\rho')))$, which implies $M(v_i) = M(u_i)$ i.e. $C(v_i) = C(u_i)$. But $(u_i, v_i) \in E$, and this contradicts the assumption that C is a valid coloring. ■

5. Dynamic Observers

In this section we introduce *dynamic observers*. To illustrate why dynamic observers can be useful consider the following example.

Example 3 (Dynamic Observation) *Assume we want to detect faults in automaton $\mathcal{B}$ of Figure 3. A static diagnoser that observes $\Sigma = \{a, b\}$ works, however, no proper subset of Σ can be used to detect faults in $\mathcal{B}$. Thus the minimum cardinality of the set of observable events for diagnosing $\mathcal{B}$ is 2 i.e. a static observer will have to monitor two events during the execution of the DES. If we want to use a mask, the minimum-cardinality for a mask is 2 as well. This means that an observer will have to be receptive to at least two inputs at each point in time to detect a fault in $\mathcal{B}$. One can think of being receptive as switching on a device to sense an event. This consumes energy. We can be more efficient using a dynamic observer, that only turns on sensors when needed, thus saving energy. In the case of $\mathcal{B}$, this can be done as follows: in the beginning we only switch on the a-sensor; once an a occurs the a-sensor is switched off and the b-sensor is switched on. Compared to the previous diagnosers we use half as much energy.*

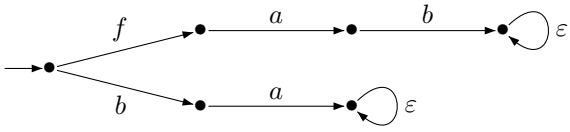


Figure 3. The automaton $\mathcal{B}$

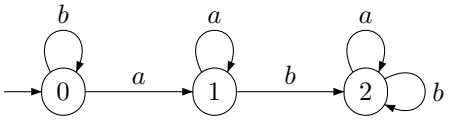
5.1. Dynamic Observers

We formalize the above notion of dynamic observation using *observers*. The choice of the events to observe can depend on the choices the observer has made before and on the observations it has made. Moreover an observer may have *unbounded* memory.

Definition 4 (Observer) *An* observer Obs *over Σ is a deterministic labeled automaton* Obs $= (S, s_0, \Sigma, \delta, L)$, *where S is a (possibly infinite) set of states, $s_0 \in S$ is the initial state, Σ is the set of observable events, $\delta : S \times \Sigma \rightarrow S$ is the transition function (a total function), and $L : S \rightarrow 2^{\Sigma}$ is a labeling function that specifies the set of events that the*

observer wishes to observe when it is at state s. We require for any state s and any $a \in \Sigma$, if $a \notin L(s)$ then $\delta(s,a) = s$: this means the observer does not change its state when an event it has chosen not to observe occurs. We use the notation $\delta(s_0, w)$ to denote the state s reached after reading the word w and $L(\delta(s_0, w))$ is the set of events obs observes after w. ■

An observer implicitly defines a *transducer* that consumes an input event $a \in \Sigma$ and, depending on the current state s, either outputs a (when $a \in L(s)$) and moves to a new state $\delta(s,a)$, or outputs ε, (when $a \notin L(s)$) and remains in the same state waiting for a new event. Thus, an observer defines a mapping Obs from Σ^* to Σ^* (we use the same name "Obs" for the automaton and the mapping). Given a run ρ, Obs($\boldsymbol{\pi}_{/\Sigma}(tr(\rho))$) is the output of the transducer on ρ. It is called the *observation* of ρ by Obs. We next provide an example of a particular case of observer which can be represented by a finite-state machine.



$L(0) = \{a\} \quad L(1) = \{b\} \quad L(2) = \emptyset$

Figure 4. A finite-state observer Obs

Example 4 *Let Obs be the observer of Figure 4 that maps the following inputs as follows: Obs$(baab) = ab$, Obs$(bababbaab) = ab$, Obs$(bbbbba) = a$ and Obs$(bbaaa) = a$. If Obs operates on the DES $\mathcal{B}$ of Figure 3 and $\mathcal{B}$ generates $f.a.b$, Obs will have as input $\boldsymbol{\pi}_{/\Sigma}(f.a.b) = a.b$ with $\Sigma = \{a, b\}$. Consequently the observation of Obs is Obs$(\boldsymbol{\pi}_{/\Sigma}(f.a.b)) = a.b$.*

5.2. Fault Diagnosis with Dynamic Diagnosers

Definition 5 ((Obs, k)-diagnoser) *Let A be a finite automaton over $\Sigma^{\varepsilon,f}$ and* Obs *be an observer over Σ. $D : \Sigma^* \to \{0,1\}$ is an* (Obs, k)-diagnoser *for A if (i) $\forall \rho \in$ NonFaulty(A), $D(\text{Obs}(\boldsymbol{\pi}_{/\Sigma}(tr(\rho)))) = 0$ and (ii) $\forall \rho \in$ Faulty$_{\geq k}(A)$, $D(\text{Obs}(\boldsymbol{\pi}_{/\Sigma}(tr(\rho)))) = 1$.* ■

A is (Obs, k)-diagnosable if there is an (Obs, k)-diagnoser for A. A is Obs-diagnosable if there is some k such that A is (Obs, k)-diagnosable.

If a diagnoser always selects Σ as the set of observable events, it is a static observer and (Obs, k)-diagnosability amounts to the standard (Σ, k)-diagnosis problem [9]. In this case A is (Σ, k)-diagnosable iff equation (1) holds.

As for Σ-diagnosability, we have the following equivalence for dynamic observers: A is (Obs, k)-diagnosable iff $\text{Obs}(\boldsymbol{\pi}_{/\Sigma}(\text{Faulty}^{tr}_{\geq k}(A))) \cap \text{Obs}(\boldsymbol{\pi}_{/\Sigma}(\text{NonFaulty}^{tr}(A))) = \emptyset$. This follows directly from definition 5.

Problem 3 (Finite-State Obs-Diagnosability)
INPUT: A, Obs *a finite-state observer.*
PROBLEM:

(A) Is A Obs*-diagnosable?*

(B) If the answer to (A) is "yes", compute the minimum k such that A is (Obs, k)-diagnosable.

Theorem 3 *Problem 3 is in P.*

Proof: (Sketch) The proof runs as follows: we build a *product* automaton[3] $A \otimes \text{Obs}$ such that: A is (Obs, k)-diagnosable $\iff$ $A \otimes \text{Obs}$ is (Σ, k)-diagnosable. Let $A = (Q, q_0, \Sigma^{\varepsilon,f}, \to)$ be a finite automaton and Obs $= (S, s_0, \Sigma, \delta, L)$ be a finite-state observer. We define the automaton $A \otimes \text{Obs} = (Q \times S, (q_0, s_0), \Sigma^{\varepsilon,f}, \to)$ as follows:

- $(q,s) \xrightarrow{\beta} (q', s')$ iff $\exists \lambda \in \Sigma$ s.t. $q \xrightarrow{\lambda} q'$, $s' = \delta(s, \lambda)$ and $\beta = \lambda$ if $\lambda \in L(s)$, $\beta = \varepsilon$ otherwise;
- $(q,s) \xrightarrow{\lambda} (q', s)$ iff $\exists \lambda \in \{\varepsilon, f\}$ s.t. $q \xrightarrow{\lambda} q'$.

As the size of $A \otimes \text{Obs}$ is polynomial in the size of A and Obs the result follows. ■

For Problem 3, we have assumed that an observer was given. It would be even better if we could *synthesize* an observer Obs such that the plant A is diagnosable. Before attempting to synthesize such an observer, we should first check that the plant is Σ-diagnosable: if it is not, then obviously no such observer exists; if the plant is Σ-diagnosable, then the trivial observer that observes all events in Σ at all times works[4]. Therefore, we need a way to compute all the "good" observers. Hence we define the problem of computing the set of *all* valid observers.

Problem 4 (Dynamic-Diagnosability)
INPUT: A.
PROBLEM: *Compute the set $\mathcal{O}$ of all observers such that A is* Obs*-diagnosable iff* Obs $\in \mathcal{O}$.

We do not have a solution to the above problem: it can be reduced to finding a trace-based winning strategy for a Büchi game with partial observation. To do this, we use a Büchi objective instead of a safety objective in the construction given in section 5.3. We know how to do this for safety games (Appendix B in [1]) but we do not have a solution to solve Büchi games of this type. Instead, we introduce a restricted variant:

[3] We use $\otimes$ to clearly distinguish this product from the synchronous product $\times$.

[4] Notice that this also shows that existence of an observer implies existence of a finite-state observer, since the trivial observer is finite-state.

Problem 5 (Dynamic-k-Diagnosability)
INPUT: A, $k \in \mathbb{N}$.
PROBLEM: *Compute the set $\mathcal{O}$ of all observers such that A is (Obs, k)-diagnosable iff* $\text{Obs} \in \mathcal{O}$.

5.3. Problem 5 as a Game Problem

To solve Problem 5 we reduce it to a *safety* 2-player game. The definitions and results for such games are given in appendix B in [1]. We also provide an intuitive explanation of such games in this section, as we construct our reduction proof. In short, the reduction we propose is the following:

- Player 1 chooses the set of events it wishes to observe, then it hands over to Player 2;
- Player 2 chooses an event and tries to produce a run which is the observation of a k-faulty run and a non-faulty run.

Player 2 wins if he can produce such a run. Otherwise Player 1 wins. Player 2 has complete information of Player 1's moves (i.e. , it can observe the sets that Player 1 chooses to observe). Player 1, on the other hand, only has partial information of Player 2's moves because not all events are observable (details follow). Let $A = (Q, q_0, \Sigma^{\varepsilon,f}, \rightarrow)$ be a finite automaton. To define the game, we use two copies of automaton A: A_1^k and A_2. The accepting states of A_1^k are those corresponding to runs of A which are faulty and where more than k steps occurred after the fault. A_2 is a copy of A where the f-transitions have been removed. The game we are going to play is the following:

1. the game starts in an state (q_1, q_2) corresponding to the initial state of the product of A_1^k and A_2. Initially, it is Player 1's turn to play. Player 1 chooses a set of events he is going to observe i.e. a subset X of Σ and hands it over to Player 2;
2. assume the automata A_1^k and A_2 are in states (q_1, q_2). Player 2 can change the state of A_1^k and A_2 by:
   (a) firing an action which is not in X in either A_1^k or A_2 (no synchronization). In this case a new state (q, q') is reached and Player 2 can play again from this state;
   (b) firing an action λ in X: to do this both A_1^k and A_2 must be in a state where λ is possible (synchronization); after the action is fired a new state (q_1', q_2') is reached: now it is Player 1's turn to play, and the game continues as in step 1 above from the new state (q_1', q_2').

Player 2 wins if he can reach a state (q_1, q_2) in $A_1^k \times A_2$ where q_1 is an accepting state of A_1^k (this means that Player 1 wins if it can avoid ad infinitum this set of states). In this sense this is a safety game for Player 1 (and a reachability game for Player 2).

Formally, the game $G_A = (S_1 \uplus S_2, s_0, \Sigma_1 \uplus \Sigma_2, \delta)$ is defined as follows ($\uplus$ denotes union of disjoint sets):

- $S_1 = (Q \times \{-1, \cdots, k\}) \times Q$ is the set of Player 1 states; a state $((q_1, j), q_2) \in S_1$ indicates that A_1^k is in state q_1, j steps have occurred after a fault, and q_2 is the current state of A_2. If no fault has occurred, $j = -1$ and if more than k steps occurred after the fault, we use $j = k$.
- $S_2 = (Q \times \{-1, \cdots, k\}) \times Q \times 2^{\Sigma}$ is the set of Player 2 states. For a state $((q_1, j), q_2, X) \in S_2$, the triple $((q_1, j), q_2)$ has the same meaning as for S_1, and X is the set of moves Player 1 has chosen to observe on its last move.
- $s_0 = ((q_0, -1), q_0)$ is the initial state of the game belonging to Player 1;
- $\Sigma_1 = 2^{\Sigma}$ is the set of moves of Player 1; $\Sigma_2 = \Sigma^{\varepsilon}$ is the set of moves of Player 2 (as we encode the fault into the state, we do not need to distinguish f from ε).
- the transition relation $\delta \subseteq (S_1 \times \Sigma_1 \times S_2) \cup (S_2 \times \{\varepsilon\} \times S_2) \cup (S_2 \times \Sigma \times S_1)$ is defined by:
  - Player 1 moves: let $\sigma \in \Sigma_1$ and $s_1 \in S_1$. Then $(s_1, \sigma, (s_1, \sigma)) \in \delta$.
  - Player 2 moves: a move of Player 2 is either a silent move (ε) i.e. a move of A_1^k or A_2 or a joint move of A_1^k and A_2 with an observable action in X. Consequently, a *silent* move $((q_1, i), q_2, X), \varepsilon, (q_1', j), q_2', X))$ is in δ if one of the following conditions holds:
    1. either $q_2' = q_2$, $q_1 \xrightarrow{\ell} q_1'$ is a step of A_1^k, $\ell \notin X$, and if $i \geq 0$ then $j = \min(i+1, k)$; if $i = -1$ and $\ell = f$ $j = 0$ otherwise $j = i$.
    2. either $q_1' = q_1$, $q_2 \xrightarrow{\ell} q_2'$ is a step of A_2, $\ell \notin X$ (and $\ell \neq f$), and if $i \geq 0$ then $j = \min(i+1, k)$, otherwise $j = i$.

    A *visible* move can be taken by Player 2 if both A_1^k and A_2 agree on doing such a move. In this case the game proceeds to a Player 1 state: $((q_1, i), q_2, X), \ell, ((q_1', j), q_2')) \in \delta$ if $\ell \in X$, $q_1 \xrightarrow{\ell} q_1'$ is a step of A_1^k, $q_2 \xrightarrow{\ell} q_2'$ is a step of A_2, and if $i \geq 0$ then $j = \min(i+1, k)$, otherwise $j = i$.

We can show that for any observer O s.t. A is (O,k)-diagnosable, there is a strategy $f(O)$ for Player 1 in G_A s.t. $f(O)$ is *trace-based* and winning. A *strategy* for Player 1 is a mapping $f : \mathit{Runs}(G_A) \to \Sigma_1$ that associates a move $f(\rho)$ in Σ_1 to each run ρ in G_A that ends in an S_1-state. A strategy f is trace-based (see appendix B in [1] for details), if given two runs ρ, ρ', if $\mathit{tr}(\rho) = \mathit{tr}(\rho')$ then $f(\rho) = f(\rho')$. Conversely, for any trace-based winning strategy f (for Player 1), we can build an observer $O(f)$ s.t. A is $(O(f),k)$-diagnosable.

Let $O = (S, s_0, \Sigma, \delta, L)$ be an observer for A. We define the strategy $f(O)$ on finite runs of G_A ending in a Player 1 state by: $f(O)(\rho) = L(\delta(s_0, \boldsymbol{\pi}_{/\Sigma}(\mathit{tr}(\rho))))$. The intuition is that we take the run ρ in G_A, take the trace of ρ (choices of Player 1 and moves of Player 2) and remove the choices of Player 1. This gives a word in Σ^*. The strategy for Player 1 for ρ is the set of events the observer O chooses to observe after reading $\boldsymbol{\pi}_{/\Sigma}(\mathit{tr}(\rho))$ i.e. $L(\delta(s_0, \boldsymbol{\pi}_{/\Sigma}(\mathit{tr}(\rho))))$.

Theorem 4 *Let O be an observer s.t. A is (O,k)-diagnosable. Then $f(O)$ is a trace-based winning strategy in G_A.*

Proof: First $f(O)$ is trace-based by definition. We have to prove that $f(O)$ is winning. We denote $Out(G,f)$ the set of outcomes i.e. the set of possible runs of a game G when the strategy f is played by Player 1 (see Appendix B in [1] for a formal definition of $Out(G,f)$). Assume on the contrary that $f(O)$ is not winning. This implies that there is a run ρ in $Out(G_A, f(O))$ as defined by equations (2–5). Each step i of the run given by one of equations (2–5) consists of a choice of Player 1 (X_i move) followed by a number of moves by Player 1 (λ_i^j actions). The last state encountered in ρ, $((q_n^1(\alpha), k_n(\alpha)), q_n^2(\alpha), X_n)$ is a losing state for Player 1, which means that $k_n(\alpha) \geq k$, by definition of losing states in G_A. From the run ρ, we can build two runs ν and ν' defined by equations (6) and (7). By definition of G_A, each λ_i^j is either a common visible action of A_1^k and A_2 and it is in Σ, or a silent action (ε) i.e. it comes from an action of A_1^k or A_2 that is not in the current set of visible actions X_i. We remove from ν (resp. ν') the actions ε that are obtained from an action of A_2 (resp. A_1^k) leaving the state of A_1^k (resp. A_2) unchanged. Let $\tilde{\nu}$ and $\tilde{\nu}'$ be the runs obtained this way. By definition of G_A, $\tilde{\nu} \in \mathit{Faulty}_{\geq k}(A)$ and $\tilde{\nu}' \in \mathit{NonFaulty}(A)$. We claim that $O(\tilde{\nu}) = O(\tilde{\nu}')$. Indeed, each part of the runs from $q_i^1 \cdots q_{i+1}^1$ and $q_i^2 \cdots q_{i+1}^2$ yields the same observation by O: it is the sequence of events $\lambda_{j_1} \cdots \lambda_{j_{n_i}}$ s.t. each λ_{j_l} is a letter of both A_1^k and A_2 and is in X_i. As there are two runs $\tilde{\nu} \in \mathit{Faulty}_{\geq k}(A)$ and $\tilde{\nu}' \in \mathit{NonFaulty}(A)$ with the same observation, A is not (O,k)-diagnosable which contradicts the assumption. Hence $f(O)$ must be winning. ■

Conversely, with each trace-based strategy f of the game G_A we can associate an automaton $O(f) = (S, s_0, \Sigma, \delta, L)$ defined by:

- $S = \{\boldsymbol{\pi}_{/\Sigma}(\mathit{tr}(\rho)) \mid \rho \in \mathit{Out}(G_A, f) \text{ and } \mathit{tgt}(\rho) \in S_1\}$;
- $s_0 = \varepsilon$;
- $\delta(v, \ell) = v'$ if $v \in S$, $v' = v.\ell$ and there is a run $\rho \in \mathit{Out}(G_A, f)$ with $\rho = q_0 \xrightarrow{X_0} q_0^1 \xrightarrow{\varepsilon^*} q_0^{n_0} \xrightarrow{\lambda_1} q_1 \xrightarrow{X_1} q_1^1 \xrightarrow{\varepsilon^*} q_1^{n_1} \xrightarrow{\lambda_2} q_2 \cdots q_{k_1} \xrightarrow{\varepsilon^*} q_{k-1}^{n_{k-1}} \xrightarrow{\lambda_k} q_k$ with each $q_i \in S_1$, $q_i^j \in S_2$, $v = \boldsymbol{\pi}_{/\Sigma}(\mathit{tr}(\rho))$, and $\rho \xrightarrow{X_k} q_k^1 \xrightarrow{\varepsilon^*} q_k^{n_k} \xrightarrow{\ell} q_{k+1}$ with $q_{k+1} \in S_1$, $\ell \in X_k$. $\delta(v, l) = v$ if $v \in S$ and $\ell \notin f(\rho)$;
- $L(v) = f(\rho)$ if $v = \boldsymbol{\pi}_{/\Sigma}(\mathit{tr}(\rho))$.

Lemma 1 *$O(f)$ is an observer.*

Proof: We first have to prove that $O(f)$ (more precisely L) is well defined. Assume $v = \boldsymbol{\pi}_{/\Sigma}(\mathit{tr}(\rho))$ and $v' = \boldsymbol{\pi}_{/\Sigma}(\mathit{tr}(\rho'))$. As f is trace-based, $f(\rho) = f(\rho')$ and there is a unique value for $L(v)$.

We also have to prove that the last requirement of Definition 4 is satisfied i.e. if $a \notin L(s)$ then $\delta(s,a) = s$. If $\ell \notin L(v)$, then $\ell \notin f(\boldsymbol{\pi}_{/\Sigma}(\mathit{tr}(\rho)))$ for any ρ s.t. $v = \boldsymbol{\pi}_{/\Sigma}(\mathit{tr}(\rho))$ because f is trace-based. Thus $\delta(v, \ell) = v$. ■

Theorem 5 *Let f be a trace-based winning strategy in G_A. Then A is $(O(f),k)$-diagnosable.*

Proof: Assume A is not $(O(f),k)$-diagnosable. There are two runs $\nu \in \mathit{Faulty}_{\geq k}(A)$ and $\nu' \in \mathit{NonFaulty}(A)$ s.t. $O(f)(\boldsymbol{\pi}_{/\Sigma}(\mathit{tr}(\nu))) = O(f)(\boldsymbol{\pi}_{/\Sigma}(\mathit{tr}(\nu')))$. Let $\tilde{\nu}$ (resp. $\tilde{\nu'}$) be the sequence of labels that appear in ν (resp. ν'). We can write $\tilde{\nu}$ and $\tilde{\nu'}$ in the form $\tilde{\nu} = w_{-1}\lambda_0 w_0 \lambda_1 w_1 \cdots \lambda_n w_n$ and $\tilde{\nu'} = w'_{-1}\lambda_0 w'_0 \lambda_1 w'_1 \cdots \lambda_n w'_n$ with $w_i, w'_i \in (\Sigma \setminus O(f)(\lambda_0\lambda_1\cdots\lambda_i)^*$ for $i \geq 0$ and $w_{-1}, w'_{-1} \in (\Sigma \setminus O(f)(\varepsilon))^*$, and $\lambda_{i+1} \in O(f)(\lambda_0\lambda_1\cdots\lambda_i)$. We build a run in $Out(G_A, f)$ as follows:

1. Player 1 chooses the set $X_0 = O(f)(\varepsilon)$ which is by definition equal to $f((q_0^1, -1), q_0^2)$ where $((q_0^1, -1), q_0^2)$ is the initial state of the game.
2. Player 2 plays the sequence of actions in w_1 and w'_1 synchronizing on the common actions of w_1 and w'_1. The game moves through S_2 states because each action is an invisible move. Finally Player 2 chooses $\lambda_0 \in O(f)(\varepsilon)$. The game reaches a new S_1-state $((q_1^1, k_1), q_1^2)$.
3. from $((q_1^1, k_1), q_1^2)$, the strategy f is to play X_1 which by definition is $O(f)(\lambda_0)$. Thus Player 2 can play the sequence of actions given in w_2 and w'_2 synchronizing again on common action. In the end Player 2 plays $\lambda_1 \in O(f)(\lambda_0)$.

$$\rho = (q_0^1,-1),q_0^2 \xrightarrow{X_0} (q_0^1,0),q_0^2,X_0 \xrightarrow{\lambda_0^1} (q_0^1(1),k_0(1)),q_0^2(1),X_0\cdots(q_0^1(j),k_0(j)),q_0^2(j),X_0\cdots \xrightarrow{\lambda_0^{n_0}} \quad (2)$$

$$(q_1^1,k_1),q_1^2 \xrightarrow{X_1} (q_1^1,k_1),q_1^2,X_1 \xrightarrow{\lambda_1^1} (q_1^1(1),k_1(1)),q_1^2(1),X_1\cdots(q_1^1(j),k_1(j)),q_1^2(j),X_1\cdots \xrightarrow{\lambda_1^{n_1}} \quad (3)$$

$$\vdots \quad (4)$$

$$(q_n^1,k_n),q_n^2 \xrightarrow{X_n} (q_n^1,k_n),q_n^2,X_n\cdots(q_n^1(j),k_n(j)),q_n^2(j),X_n\cdots \xrightarrow{\lambda_0^{\alpha}} (q_n^1(\alpha),k_n(\alpha)),q_n^2(\alpha),X_n \quad (5)$$

$$\nu = q_0^1 \xrightarrow{\lambda_0^1} q_0^1(1) \xrightarrow{\lambda_0^2} \cdots \xrightarrow{\lambda_0^{n_0}} q_1^1 \xrightarrow{\lambda_1^1} \cdots \xrightarrow{\lambda_1^{n_1}} q_2^1\cdots q_n^1 \xrightarrow{\lambda_n^1} \cdots \xrightarrow{\lambda_n^{\alpha}} q_n^1(\alpha) \quad (6)$$

$$\nu' = q_0^2 \xrightarrow{\lambda_0^1} q_0^2(1) \xrightarrow{\lambda_0^2} \cdots \xrightarrow{\lambda_0^{n_0}} q_1^2 \xrightarrow{\lambda_1^1} \cdots \xrightarrow{\lambda_1^{n_1}} q_2^2\cdots q_n^2 \xrightarrow{\lambda_n^1} \cdots \xrightarrow{\lambda_n^{\alpha}} q_n^2(\alpha) \quad (7)$$

We can iterate the previous algorithm and build a run in $Out(G_A,f)$ that reaches a state $((q_n^1,k_n),q_n^2)$ with $k_n \geq k$ and thus $Out(G_A,f)$ contains a losing run. Hence f is not winning which contradicts the assumption. This way we conclude that A is $(O(f),k)$-diagnosable. ■

The result on G_A (Appendix B in [1]) is that, if there is a winning trace-based strategy for Player 1, then there is a most permissive strategy $\mathcal{F}_A$ which has finite memory. It can be represented by a finite automaton $S_{\mathcal{F}_A} = (W_1 \uplus W_2, s_0, \Sigma \cup 2^{\Sigma}, \Delta_A)$ s.t. $\Delta_A \subseteq (W_1 \times 2^{\Sigma} \times W_2) \cup (W_2 \times \Sigma \times W_1)$ which has size exponential in the size of G_A. For a given run $\rho \in (\Sigma \cup 2^{\Sigma})^*$ ending in a W_1-state, we have $\mathcal{F}_A(w) = \mathit{en}(\Delta_A(s_0,w))$.

5.4. Most Permissive Observer

We now define the notion of a most *permissive* observer and show the existence of a most permissive observer for a system in case A is diagnosable. $\mathcal{F}_A$ is the mapping defined at the end of the previous section.

For an observer $O = (S,s_0,\Sigma,\delta,L)$ and $w \in \Sigma^*$ we let $L(w)$ be the set $L(\delta(s_0,w))$: this is the set of events O chooses to observe on input w. Given a word $\rho \in \boldsymbol{\pi}_{/\Sigma}(\mathcal{L}(A))$, we recall that $O(\rho)$ is the observation of ρ by O. Assume $O(\rho) = a_0\cdots a_k$. Let $\overline{\rho} = L(\varepsilon).\varepsilon.L(a_0).a_0.\cdots L(O(\rho)(k)).a_k$ i.e. $\overline{\rho}$ contains the history of what O has chosen to observe at each step and the events that occurred after each choice.

Let $\mathcal{O} : (2^{\Sigma} \times \Sigma^{\varepsilon})^+ \to 2^{2^{\Sigma}}$. By definition $\mathcal{O}$ is the most permissive observer for (A,k) if the following holds:

$$\begin{array}{c} O=(S,s_0,\Sigma,\delta,L) \\ \textit{is an observer and} \\ A \textit{ is } (O,k)\textit{-diagnosable} \end{array} \iff \begin{array}{c} \forall w \in \Sigma^* \\ L(\delta(s_0,w)) \in \mathcal{O}(\overline{w}) \end{array}$$

The definition of the most permissive observer states that:

- any good observer O (one such that A is (O,k)-diagnosable) must choose a set of observable events in $\mathcal{O}(\overline{w})$ on input w;
- if an observer chooses its set of observable events in $\mathcal{O}(\overline{w})$ on input w, then it is a good observer.

Assume A is (Σ,k)-diagnosable. Then there is an observer O s.t. A is (O,k)-diagnosable because the constant observer that observes Σ is a solution. By Theorem 4, there is a trace-based winning strategy for Player 1 in G_A. As said at the end of the previous subsection, in this case there is a most permissive trace-based winning strategy which is $\mathcal{F}_A$.

Theorem 6 *$\mathcal{F}_A$ is the most permissive observer.*

Proof: Let $O = (S,s_0,\Sigma,\delta,L)$ be an observer such that A is (O,k)-diagnosable. We have to prove that $L(\delta(s_0,w)) \in \mathcal{F}_A(\overline{w})$ for any $w \in \Sigma^*$. By Theorem 4, the strategy $f(O)$ is a winning trace-based strategy and this implies that $f(O)(\nu) \in \mathcal{F}_A(\nu)$ for any run ν of G_A. By definition of $\overline{w}$, $\boldsymbol{\pi}_{/\Sigma}(\overline{w}) = w$. By definition of $f(O)$, $f(O)(\overline{w}) = L(\delta(s_0,\boldsymbol{\pi}_{/\Sigma}(\mathit{tr}(\overline{w})))) = L(\delta(s_0,w))$ and thus $L(\delta(s_0,w)) \in \mathcal{F}_A(\overline{w})$.

Conversely, assume O is such that $\forall w \in \Sigma^*, L(s_0,w) \in \mathcal{F}_A(\overline{w})$. We have to prove that A is (O,k)-diagnosable. Again, we build $f(O)$. As before, $f(O)$ is a winning trace-based strategy in G_A and thus $O(f(O))$ is such that A is $(O(f(O)),k)$-diagnosable by Theorem 5. Assume $O(f(O)) = (S',s_0',\Sigma,\delta',L'))$. By construction of $O(f(O))$, $L'(\delta'(s_0',w)) = f(O)(\rho)$ if $w = \boldsymbol{\pi}_{/\Sigma}(\mathit{tr}(\rho))$. Hence $O(f(O)) = O$ and A is (O,k)-diagnosable. ■

This enables us to solve Problem 5 and compute a finite representation of the set $\mathcal{O}$ of all observers such that A is (O,k)-diagnosable iff $O \in \mathcal{O}$.

Computing $\mathcal{F}_A$ can be done in $O(2^{|G_A|})$ (Appendix in [1]). The size of G_A is quadratic in $|A|$, linear in the size of k, and exponential in the size of Σ i.e. $|G_A| = O(|A|^2 \times 2^{|\Sigma|} \times |k|)$. This means that computing $\mathcal{F}_A$ can be done in exponential time in the size of A and k and doubly exponential time in the size of Σ.

6. Conclusion and Future Work

In this paper we have addressed sensor minimization problems in the context of fault diagnosis, using both static and dynamic observers. We showed that computing the smallest number of observable events necessary to achieve diagnosis with a static observer is NP-complete: this result also holds in the mask-based setting which allows to consider events that are observable but not distinguishable. We then focused on dynamic observers and proved that, for a given such observer, diagnosability can be checked in polynomial time (as in the case of static observers). We also solved the synthesis problem of dynamic observers and showed that a most-permissive dynamic observer can be computed in doubly-exponential time.

We are currently investigating the following directions:

- Problem 4 has not been solved so far. The major impediment to solve it is that the reduction we propose in section 5 yields a Büchi game. The algorithm we give in appendix B in [1], does not work for Büchi games and cannot be extended trivially.
- Problem 5 is solved in doubly exponential time. To reduce the number of states of the most permissive observer, we point out that only *minimal* sets of events we need to be observed. Indeed, if we can diagnose a system by observing only A from some point on, we surely can diagnose it using any superset $A' \supseteq A$. So far we keep all the sets that can be used to diagnose the system. We could possibly take advantage of the previous property using techniques described in [5].
- Another line of work is to define a notion of *cost* for dynamic observers. This can be done and an optimal observer can be computed as it is reported in [2].

Acknowledgments. The authors wish to thank the anonymous referees for their detailed and helpful comments.

References

[1] Franck Cassez, Stavros Tripakis, and Karine Altisen. Sensor minimization problems with static or dynamic observers for fault diagnosis. Technical Report RI-2007-1, IRCCyN/CNRS, 1 rue de la Noë, BP 92101, 44321 Nantes Cedes, France, January 2007. Available at `http://www.irccyn.fr/franck`.

[2] Franck Cassez, Stavros Tripakis, and Karine Altisen. Synthesis of optimal-cost dynamic diagnosers for fault diagnosis of discrete-event systems. In *1st IEEE & IFIP International International Symposium on Theoretical Aspects of Software Engineering (TASE'07)*, Shangai, ROC, July 2007. IEEE Computer Society. Forthcoming.

[3] R. Cieslak, C. Desclaux, A. Fawaz, and Pravin Varaiya. Supervisory control of discrete-event processes with partial observations. *IEEE Transactions on Automatic Control*, 33:249–260, 1988.

[4] Rami Debouk, Stéphane Lafortune, and Demosthenis Teneketzis. On an optimization problem in sensor selection. *Discrete Event Dynamic Systems*, 4(12), November 2004.

[5] Laurent Doyen, Krishendu Chatterjee, Thomas A. Henzinger, and Jean-François Raskin. Algorithms for omega-regular games with imperfect information. In *CSL: Computer Science Logic*, Lecture Notes in Computer Science 4207, pages 287–302. Springer, 2006.

[6] Shengbing Jiang, Zhongdong Huang, Vigyan Chandra, and Ratnesh Kumar. A polynomial algorithm for testing diagnosability of discrete event systems. *IEEE Transactions on Automatic Control*, 46(8), August 2001.

[7] Shengbing Jiang, Ratnesh Kumar, and Humberto E. Garcia. Optimal sensor selection for discrete event systems with partial observation. *IEEE Transactions on Automatic Control*, 48(3):369–381, March 2003.

[8] Peter Ramadge and W. Murray Wonham. Supervisory control of a class of discrete event processes. *SIAM J. Control Optim.*, 25(1), January 1987.

[9] Meera Sampath, Raja Sengupta, Stephane Lafortune, Kasim Sinnamohideen, and Demosthenis C. Teneketzis. Diagnosability of discrete event systems. *IEEE Transactions on Automatic Control*, 40(9), September 1995.

[10] John N. Tsitsiklis. On the control of discrete event dynamical systems. *Mathematics of Control, Signals and Systems*, 2(2), 1989.

[11] Tae-Sic Yoo and Stéphane Lafortune. On the computational complexity of some problems arising in partially-observed discrete event systems. In *American Control Conference (ACC'01)*, 2001. Arlington, VA.