DYNAMIC DIAGNOSIS OF ACTIVE SYSTEMS WITH FRAGMENTED OBSERVATIONS

Gianfranco Lamperti
Dipartimento di Elettronica per l'Automazione
Via Branze 38, 25123 Brescia, Italy

Marina Zanella
Dipartimento di Elettronica per l'Automazione
Via Branze 38, 25123 Brescia, Italy

Keywords: Knowledge-based systems engineering, model-based reasoning, diagnosis, discrete-event systems, active systems, communicating automata, monitoring, uncertainty

Abstract: Diagnosis of discrete-event systems (DESs) is a complex and challenging task. Typical application domains include telecommunication networks, power networks, and digital-hardware networks. Recent blackouts in northern America and southern Europe offer evidence for the claim that automated diagnosis of large-scale DESs is a major requirement for the reliability of this sort of critical systems. The paper is meant as a little step toward this direction. A technique for the dynamic diagnosis of active systems with uncertain observations is presented. The essential contribution of the method lies in its ability to cope with uncertainty conditions while monitoring the systems, by generating diagnostic information at the occurrence of each newly-received fragment of observation. Uncertainty stems, on the one hand, from the complexity and distribution of the systems, where noise may affect the communication channels between the system and the control rooms, on the other, from the multiplicity of such channels, which is bound to relax the absolute temporal ordering of the observable events generated by the system during operation. The solution of these diagnostic problems requires nonmonotonic reasoning, where estimates of the system state and the relevant candidate diagnoses may not survive the occurrence of new observation fragments.

1 INTRODUCTION

Diagnosis is the task of finding out the faults affecting a physical system given a set of symptoms gathered by observing the system itself. Model-based diagnosis is a research area in Artificial Intelligence devoted to proposing automated reasoning mechanisms and modeling primitives for performing such a task by exploiting the (structure and behavior) models of the considered physical systems. From the middle '90s some research efforts have been directed toward model-based diagnosis of DESs since discrete models are simpler to deal with than continuous ones (Fattah and Provan, 1997; Debouk et al., 2000; Lunze, 2000; Cordier and Largouët, 2001; Pencolé et al., 2001; Console et al., 2002) .

Diagnostic processing can be carried out either after an observation has been collected throughout a time interval or every time a new observable event is received. In order to distinguish the two situations we call *a posteriori diagnosis* the former task and *dynamic* (or *monitoring-based*) *diagnosis* the latter.

This paper deals with dynamic diagnosis of *active systems* (Lamperti and Zanella, 2003b). Interest on active systems was prompted by the case study of diagnosis of power transmission networks (Lamperti and Pogliano, 1997), aimed at preventing blackouts. Such networks are still a reference application domain although the notion of an active system has progressively been generalized so as to represent a large class of DESs. The concept of a *fragmented* observation taken as input by the task described in this paper is more general than any one by other authors, corresponding to an uncertain observation, as defined in (Lamperti and Zanella, 2002). The only difference between an uncertain and a fragmented observation is that the former cumulatively represents all the observable events received over a time interval while the latter represents a single (logically uncertain and/or temporally uncertain and/or source uncertain) observable event, called a *message*.

Lamperti G. and Zanella M. (2004).
DYNAMIC DIAGNOSIS OF ACTIVE SYSTEMS WITH FRAGMENTED OBSERVATIONS.
In *Proceedings of the Sixth International Conference on Enterprise Information Systems*, pages 249-261
DOI: 10.5220/0002619202490261

Past research has focused both on a posteriori diagnosis (Baroni et al., 1999) and dynamic diagnosis (Lamperti and Zanella, 2003a) of active systems. However, uncertain observations have so far been provided as input to a posteriori diagnosis while dynamic diagnosis had been fed only by completely certain observations. No contribution in the literature to monitoring and diagnosis of DESs takes into account observable events that are uncertain in nature. The purpose of the present work is to face this challenge.

In the remainder of the paper, Section 2 presents the context from which all the examples in the paper are drawn. Section 3 provides a formal definition of the class of considered systems. Section 4 defines the class of problems inherent to such systems that can be solved by the technique described in Section 5. Section 6 relates the current work to other works in the literature and concludes the paper.

2 POWER NETWORK

We consider a sample application domain involving power networks. Each transmission line is protected by two breakers that are commanded by a protection. The protection is designed to detect conditions that may be dangerous to the line. Typically, if a short circuit affects the line, the protection is expected to command the two breakers to open, so as to isolate the line from the remaining part of the network. In a simplified view, the network is represented by a series of lines, each one associated with a protection, as displayed in Fig. 1.

The figure outlines a portion of the network, that encompasses two lines, $\mathfrak{L}_1$ and $\mathfrak{L}_2$, with relevant protections, p_1 and p_2. For instance, p_2 controls $\mathfrak{L}_2$ by operating breakers b_{21} and b_{22}. In normal behavior, both breakers are expected to open when tripped by the protection. However, the protection system may exhibit an abnormal (faulty) behavior, for example, one breaker or both may not open when required. In such a case, each faulty breaker informs the protection about its own misbehavior. Then, the protection sends a request of recovery actions to the neighboring protections, which will operate their own breakers appropriately. For example, if p_2 operates b_{21} and b_{22} and the former is faulty, then p_2 will send a signal to p_1, which is supposed to command the breaker on the same (left-hand) side of the faulty breaker b_{21}, namely b_{11}. If both b_{21} and b_{22} are faulty, then p_2 will ask recovery actions to both the neighboring protections. The protection system is designed to propagate the recovery request until the tripped breaker opens correctly. Consequently, the greater the number of faulty breakers, the larger the extent of the subnetwork that is isolated.

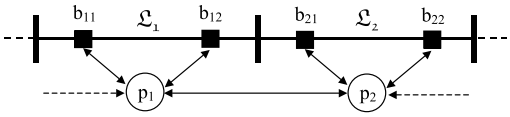

Figure 1: Power network.

When the protection system reacts to a short circuit by attempting to isolate the shorted line, possibly with the help of recovery actions, a subset of the occurring events are visible to the external world, typically to the operator of a control room who is in charge of monitoring the behavior of the network and, possibly, to perform actions so as to minimize the extent of the isolated subnetwork by means of explicit telecommands. In the ideal scenario, the reaction of the protection system is correct (normal) and the shorted line is clearly identified by the pair of open breakers. So, if the operator observes that b_{21} and b_{22} are open, he or she is allowed to assume that line $\mathfrak{L}_2$ has been isolated owing to a short circuit. If the short circuit is transient (caused, for example, by a lightning), it may be the case that, once the short has extinguished, the isolated line is reconnected to the network by the operator. If such a reconnection is successful (no reaction of the protection system is triggered anew), the network keeps on being fully operating. Instead, if the short circuit is permanent (for example, due to a tree fallen on the line), the reconnection will cause a new reaction of the protection system [1].

The reconnection problem becomes harder when the reaction of the protection system is abnormal. In fact, the operator is supposed to face the additional problem of localizing the shorted line among those embodied within the isolation. Such an identification cannot be carried out by simply looking at open breakers. With reference to Fig. 1, if the open breakers are b_{11} and b_{22}, then the shorted line will be either $\mathfrak{L}_1$ or $\mathfrak{L}_2$. A static analysis of the isolation does not provide any clue on where the short circuit is located.

[1] The protection system typically operates autonomously in two steps. First, the shorted line is isolated for a few seconds in the hope of extinguishing the (transient) short circuit. Later, the line is reconnected to the network and, if the short circuit has extinguished, then the network is completely recovered, otherwise the line is isolated permanently from the system.

Assuming the occurrence of a single short circuit, the only possible claim is that one breaker is faulty, corresponding to two different scenarios. In the first scenario, $\mathfrak{L}_1$ is shorted and b_{12} is faulty, thereby requiring the intervention of b_{22}. The second scenario is symmetric: D_2 is shorted and b_{21} is faulty. In either case, one line is not shorted and might be reconnected without any risk (provided the operator knows which it is). For instance, if the shorted line is $\mathfrak{L}_1$, then $\mathfrak{L}_2$ can be reconnected to the network by opening b_{12} manually and telecommanding the closure of b_{22}.

The localization of the short circuit and the identification of the faulty breakers may be impractical in real contexts, especially when the extent of the isolation spans several lines and the operator is required to take recovery actions within stringent time constraints. On the one hand, there is the problem of observability: the observable events generated during the reaction of the protection system are generally incomplete and uncertain in nature. On the other, whatever the 'quality' of the observation, it is practically impossible for the operator to reason on the observations under stringent time constraints, so as to make consistent hypotheses on the behavior of the system and, eventually, to establish the shorted line and the faulty breakers.

The task of the operator might be dramatically improved if we would provide a tool that supports the automated reconstruction of the system reaction and the generation of the expected information, namely, the diagnosis of the system. This requires the precise definition of the class of considered systems, namely active systems, along with a specific diagnostic technique. We then show that our network can be modeled as an active system and, as such, automatically diagnosed by means of the given technique.

3 ACTIVE SYSTEMS

A system is a network of *components* that are connected to one another through *links*. Each component is completely modeled by a *communicating automaton* $\mathcal{C}$ that reacts to events either coming from the external world or from neighboring components through links. Formally, the automaton is a 7-tuple,

$$\mathcal{C} = (\mathbf{S}, \mathbf{E}_{\text{in}}, \mathbf{I}, \mathbf{E}_{\text{out}}, \mathbf{O}, \mathbf{T}, \mathbf{P}),$$

where $\mathbf{S}$ is the set of *states*, $\mathbf{E}_{\text{in}}$ the set of *input events*, $\mathbf{I}$ the set of *input terminals*, $\mathbf{E}_{\text{out}}$ the set of *output events*, $\mathbf{O}$ the set of *output terminals*, $\mathbf{T}$ the nondeterministic *transition function*,

$$\mathbf{T} : \mathbf{S} \times \mathbf{E}_{\text{in}} \times \mathbf{I} \times 2^{\mathbf{E}_{\text{out}} \times \mathbf{O}} \mapsto 2^{\mathbf{S}},$$

and $\mathbf{P}$ the *priority hierarchy*. A transition $T \in \mathbf{T}$, from state S to state S', that is triggered by event e at input terminal I, and generates events $e_1, \ldots, e_k$ at output terminals $O_1, \ldots, O_k$, respectively, is denoted by

$$T = S \xrightarrow[(e_1,O_1),\ldots,(e_k,O_k)]{(e,I)} S'.$$

The priority hierarchy is a DAG where nodes are events in $\mathbf{E}_{\text{in}} \times \mathbf{I}$, while edges denote a partial priority relationship among events. Since transitions are triggered by input events, the priority hierarchy among events implicitly defines a priority hierarchy among the transitions in $\mathbf{T}$, specifically, if T_1 has higher priority than T_2, T_1 will be fired before T_2. Generally speaking, among the set of triggerable transitions, the actual fired transition will be one among those with highest priority.

Links, which are the means to store the events exchanged between components, are modeled by a triple

$$\mathcal{L} = (I, O, M),$$

where I is the *input terminal*, O the *output terminal*, and M the *event management*. The latter establishes the internal structure of the link and the effect of each newly inserted event. Given a link L, the function $\mathit{Ready}(L)$ returns the set of (ready) events stored in L that can be consumed in the current state of the link. For example, if the event management is a queue, $\mathit{Ready}(L)$ will be the first event in the queue. By contrast, if M is a stack, $\mathit{Ready}(L)$ will be the last-inserted event. With a more sophisticated management where priorities are defined among events stored in L, $\mathit{Ready}(L)$ is bound to return several consumable events. The priority hierarchy $\mathbf{P}$ of a component and the event management M of a link L are different, yet related, concepts: the priority relationships in $\mathbf{P}$ are applied to those transitions that are triggered by the events in $\mathit{Ready}(L)$, the latter depending on M. Since the input terminals of a component may be connected with a set $\mathbf{L}_{\text{c}}$ of links, the whole set of consumable events is denoted by $\mathit{Ready}(\mathbf{L}_{\text{c}})$, corresponding to the union of the ready events of each link in $\mathbf{L}_{\text{c}}$. Formally, a system Σ is a triple

$$\Sigma = (\mathbf{C}, \mathbf{L}, \mathbf{G}),$$

where $\mathbf{C}$ is the set of components, $\mathbf{L}$ the set of links, and $\mathbf{G}$ the *global priority hierarchy*. The latter is a DAG similar to the priority hierarchy $\mathbf{P}$ of a component model, where the involved events are pertinent to the whole system rather than to

single components. In other words, **G** enriches the DAG obtained by the union of the single **P**'s with additional precedence relationships among the whole set of events in Σ.

Example 1 Displayed in Fig. 2 are the models *Breaker* (top) and *Protection* (bottom), relevant to the protection system outlined in Fig. 1. Each model is depicted by the set of terminals (shaded box) and the communicating automaton (graph). Each input terminal is depicted as a triangle, while each output terminal O is represented as a bullet. The automaton relevant to the breaker incorporates two states, marked by 0 (closed) and 1 (open), respectively, and five transitions, namely $T_1 \cdots T_5$, represented as arrows between states. When the breaker is closed (state 0), either transition T_1 or T_3 is nondeterministically triggered by event *op* on input terminal I. T_1 moves the breaker to state 1 (closed) without generating any output event. T_3, instead, keeps the state of the breaker unchanged (open), whilst generating event f at output terminal O. Intuitively, this is an abnormal transition (T_3 is depicted as a dotted arrow), as the breaker is supposed to open when triggered, which is not the case for T_3. Event *cl* is meant to close the breaker. When in state 1 (open), such an event triggers either T_2 (normal behavior) or T_4 (faulty behavior). Note how the same event triggers transition T_5 in state 0, leaving the state unchanged (the breaker was closed already). No priority relationships are assumed for the breaker (**P** is empty).

The model of the protection embodies four input terminals, $I_1 \cdots I_4$, and four output terminals, $O_1 \cdots O_4$. Terminals O_1 and I_1 are meant for connection with the breaker on the left of the line, while terminals O_2 and I_2 are for the communication with the breaker on the right (see Fig. 1). Instead I_3 and O_3 allow the protection to exchange events with the neighboring protection on the left. The same applies to I_4 and O_4, which are a means to communicate with the adjacent protection on the right. The corresponding automaton involves four states, marked by $0 \cdots 3$, and ten transitions, $T_1 \cdots T_{10}$. State 0 stands for normal condition. The occurrence of a short circuit on the protected line is signaled by event *sh* from the standard input *In*, which triggers transition T_1. Such a transition moves the protection to state 1 by generating event *op* at both output terminals O_1 and O_2, thus commanding the two breakers to open. In state 1, the protection may receive event f either at terminal I_1 or I_2, meaning that the relevant breaker failed to open. This triggers either transition T_5 or T_6, respectively, each of which generates the recovery event *rc* at output terminals O_3 and O_4, respectively. The

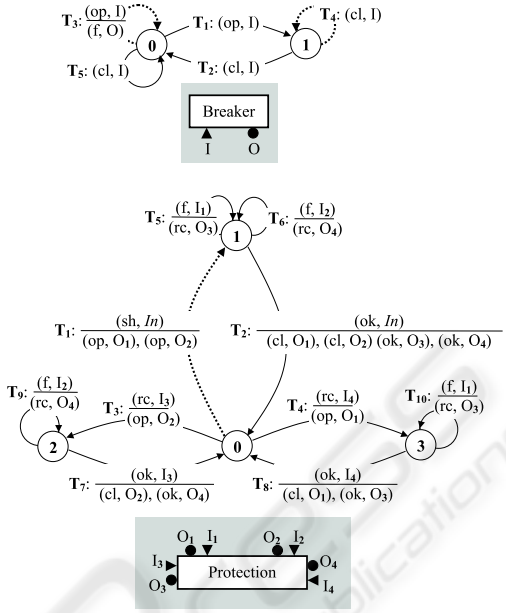

Figure 2: Component models.

extinction of the short circuit is signaled by event *ok* at the standard input terminal *In*, which triggers transition T_2 that moves the protection to state 0, while generating output events *cl* and *ok* at terminals O_1 and O_2, and O_3 and O_4, respectively. When a protection receives a request of recovery from a neighboring protection, it follows a transition from state 0 to either 2 or 3, depending on whether the request comes from the left (T_3) or from the right (T_4), respectively. So, input event (rc, I_3) makes T_3 to generate (op, O_2), that is, a command to the breaker on the right. In state 2, since even this breaker may in turn be faulty, the occurrence of event (f, I_2) triggers transition T_9 that, similarly to T_6, propagates the recovery request to the right-hand side protection. When event *ok* is received at terminal I_3 from the neighboring protection, transition T_7 moves the protection to state 0, while generating *cl* and *ok* at terminals O_2 and O_4, respectively. A symmetric behavior is defined in state 3. We assume for the protection a priority hierarchy such that (f, I_1) precedes both (ok, I_4) and (ok, In), while both (rc, I_3) and (rc, In) precede (sh, In).

Depicted on the bottom of Fig. 3 is the topology of a system Ψ that integrates the three components protecting a generic line, namely protection p and breakers b_1 and b_2. The protection is connected with the breakers by means of links $L_1 \cdots L_4$. We assume that all links share the same model, where the event management M is a simple queue. System Ψ is the abstraction of

a subpart of the protection system. Larger subparts of the network may be assembled by connecting instances of Ψ together by means of links among protections. The global priority hierarchy $\mathbf{G}$ of Ψ is such that (cl, I) of both breakers precedes (rc, I_3), (sh, In), and (rc, I_4), while (op, I) of both breakers precedes (ok, I_3), (ok, In), (ok, I_4), (f, I_1), and (f, I_2). □

3.1 Behavior Space

Given an initial state Σ_0, system Σ evolves in a way that is both consistent with its topology and the behavioral models of its components and links. The set of all possible evolutions can be thought of as the behavioral model of Σ starting at Σ_0. The resulting graph is an automaton called *behavior space*, $Bhv(\Sigma, \Sigma_0)$. A (possibly empty) path between two nodes of the space is a *history segment* of Σ. In particular, if the starting node is the initial state of the space, then such a path is a *history* of Σ[2].

Example 2 Shown in Fig. 3 is the behavior space relevant to system Ψ, where the initial state is $\Psi_0 = (\mathbb{S}_0, \mathbb{L}_0)$, where $\mathbb{S}_0 = (0, 0, 0)$ and $\mathbb{L}_0$ involves empty links. In each node, the record $\mathbb{S}$ of the component states for b_1, p, and b_2 is on the top, while the record $\mathbb{L}$ of queues of events within links $L_1 \cdots L_4$ is on the bottom. Incidentally, within the behavior space, at most one event is stored in each link, so that the state of the link can be expressed by either the label of the event or a dash, the latter denoting the empty link. Labels o and c are a shorthand for op and cl, respectively. Nodes are marked by numbers $0 \cdots 45$. For instance, in node 7 both breakers are closed (state 0 of the breaker model), while the protection has commanded the breakers to open (state 1 of the protection model). Besides, links L_1 and L_4 are empty; instead, L_2 incorporates event f (meaning that b_1 has failed to open) while L_3 contains event op, meaning that b_2 has not yet reacted to the protection command. Each edge is marked by a label identifying a component transition. Specifically, single digits refer to transitions of the protection, while two-digit strings correspond to breaker transitions. For example, 3, 31, and 42 stand for $T_3(p)$, $T_3(b_1)$, and $T_4(b_2)$, respectively. Note how Ψ becomes reacting upon the occurrence of an external event, either from the standard input (triggering transition $T_1(p)$) or from a dangling terminal (triggering either $T_3(p)$ or $T_4(p)$).

[2]The behavior space is introduced for formal reasons, but never explicitly generated by the diagnostic technique.

A history $h(\Psi)$ is identified by the sequence of labels (component transitions) marking the edges on such a path, as for instance, $h(\Psi) = \langle 1, 31, 12, 5 \rangle$, corresponding to the following scenario: (i) a short circuit occurs on the line protected by Ψ and protection p commands both breakers b_1 and b_2 to open, (ii) breaker b_1 fails to open, while (iii) breaker b_2 opens correctly, and (iv) protection p asks the neighboring protection on the left a recovery action. Finally, note the cyclicity of $Bhv(\Psi, \Psi_0)$, which means that the set of possible histories of Ψ is unbounded. □

4 DIAGNOSTIC PROBLEM

The ultimate task of diagnosis is the solution of diagnostic problems. Within the domain of active systems and the context of dynamic diagnosis, a diagnostic problem concerns the operation of the system and its solution is essentially based on the model of the system and some clues on the system reaction. Such a solution is a set of *candidate diagnoses*, each diagnosis being a set of faults relevant to a possible evolution of the system.

Formally, a *fragmented diagnostic problem* $\wp$ for a system Σ is a 4-tuple

$$\wp(\Sigma) = (\Sigma_0, \mathcal{V}, \mathcal{O}, \mathcal{R})$$

where Σ_0 is the *initial state* of Σ, that is, the state of Σ when it starts operating, $\mathcal{V}$ is the *viewer*, with specific visibility properties on the behavior of Σ, $\mathcal{O}$ is the *fragmented observation* of Σ gathered while Σ is operating, and $\mathcal{R}$ is the *ruler*, which establishes what behavior of Σ is to be considered faulty and the granularity of the diagnosis.

4.1 Viewer

A viewer establishes what component transitions are somewhat visible, as well as the specific observable label for each of them. Let $\mathbf{T}$ be the set of transitions relevant to components in Σ, and $\mathbf{V}$ a set of labels including the *null* label ε. A viewer $\mathcal{V}$ is a mapping from $\mathbf{T}$ to $\mathbf{V}$. If $(T, \varepsilon) \in \mathcal{V}$, then T is a *silent* transition, otherwise T is *visible*.

Example 3 A viewer $\mathcal{V}_\psi$ for Ψ can be defined by the set of visible transitions[3], specifically $\mathcal{V}_\psi = \{(T_1(p), sh), (T_3(p), l), (T_4(p), r), (T_1(b_1), o_1), (T_2(b_1), c_1), (T_1(b_2), o_2), (T_2(b_2), c_2)\}$. □

[3]In order to keep the examples within a reasonable complexity and without loss of generality, we assume the visibility of the short circuit by means of the *sh* label. In real application domains this is of course an over-assumption.

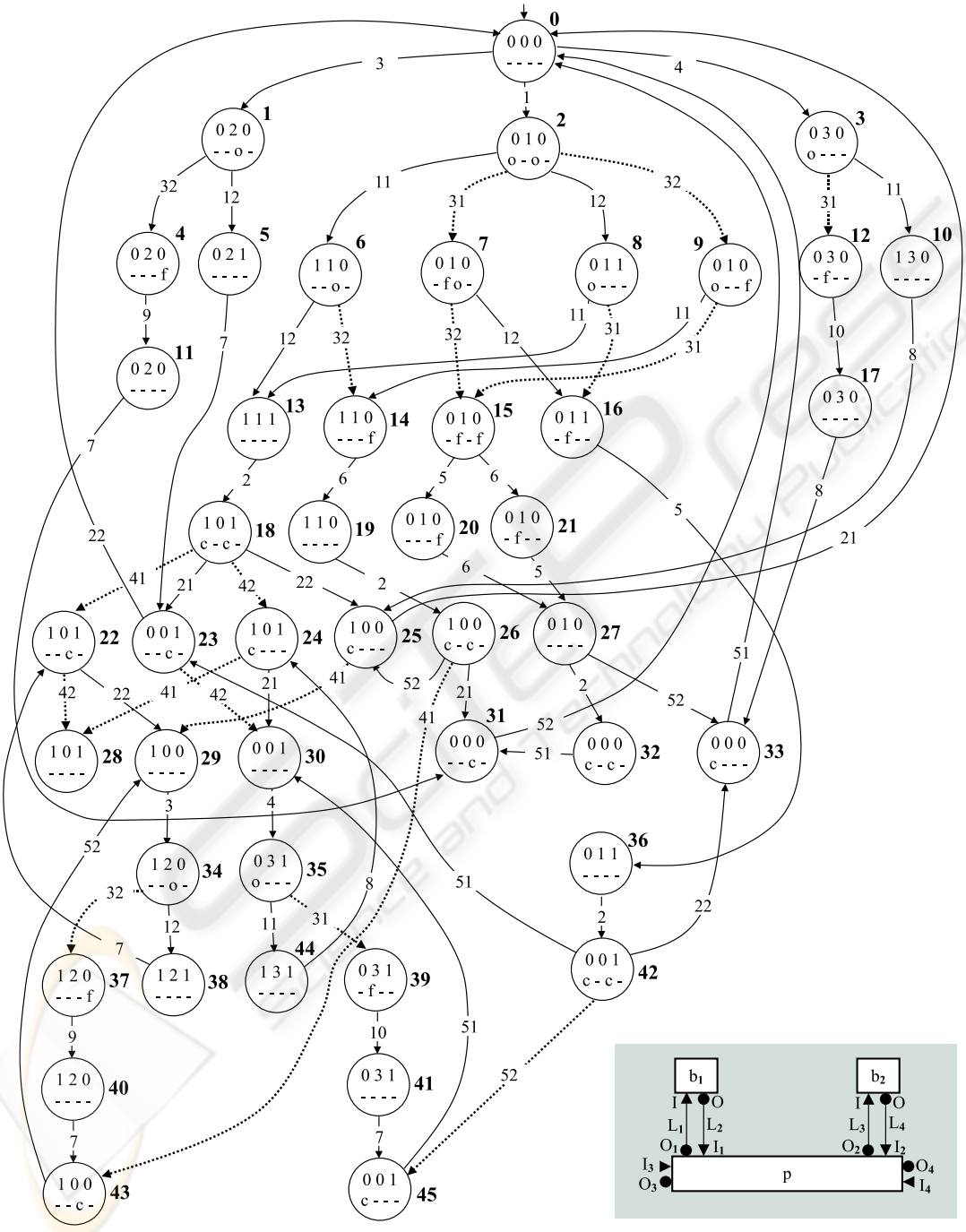

Figure 3: System Ψ (shaded) and behavior space $Bhv(\Psi, \Psi_0)$.

4.2 Fragmented Observation

When the system is operating, each visible transition is perceived by a viewer as a *message*. Each message μ is a pair (λ, τ), where $\lambda = \{\ell_1, \ldots, \ell_k\}$ is a set of labels in $\mathbf{V}$, namely the *logical content*, while $\tau = \{\mu'_1, \ldots, \mu'_h\}$ is a set of messages, namely the *temporal content*, identifying all the messages temporally preceding the current one.

A *fragmented observation* $\mathcal{O}$ is a list of messages,

$$\mathcal{O} = \langle \mu_1, \ldots, \mu_n \rangle,$$

where a *monotonicity assumption* is assumed:

$$\forall i \in [1\,..\,n], \mu_i = (\lambda_i, \tau_i)\,(\tau_i \subseteq \{\mu_1, \ldots, \mu_{i-1}\}).$$

That is, the temporal content of a message μ is supposed to refer to a (possibly empty) subset of the messages preceding μ in $\mathcal{O}$. Thus, a message is uncertain in nature, both logically and temporally. Logical uncertainty means that λ includes the actual (possibly null) label associated with the transition in $\mathcal{V}$ that generated it, but further *spurious* labels may be involved too. Temporal uncertainty means that only partial ordering is known among messages.

A fragmented observation may be mapped to an *observation graph*,

$$\gamma(\mathcal{O}) = (\Omega, \Upsilon, \Omega_0, \Omega_\mathrm{f}),$$

where Ω is the set of *nodes* isomorphic to the messages in $\mathcal{O}$, Υ is the set of *edges* isomorphic to the temporal contents of messages, $\Omega_0 \subseteq \Omega$ is the set of *roots*, and Ω_f is the set of *leaves*.

A *precedence* relationship is defined between nodes of $\gamma(\mathcal{O})$, specifically, $\omega \prec \omega'$ means that $\gamma(\mathcal{O})$ includes a path from ω to ω', while $\omega \preceq \omega'$ means either $\omega \prec \omega'$ or $\omega = \omega'$.

The graph is supposed to be in *canonical form*, that is, if $\omega \mapsto \omega'$ is an edge in Υ, then there does not exist any ω'' such that $\omega \prec \omega'' \prec \omega'$.

A *sub-observation* $\mathcal{O}_{[i]}$ of $\mathcal{O}$, where $i \in [0\,..\,n]$, is the (possibly empty) prefix of $\mathcal{O}$ up to the i-th message, namely

$$\mathcal{O}_{[i]} = \langle \mu_1, \ldots, \mu_i \rangle.$$

If the following conditions hold:

$$\mu_1 = (\{\ell_1\}, \emptyset), \forall i \in [2\,..\,n]\,(\mu_i = (\{\ell_i\}, \{\mu_{i-1}\}))\,,$$

then $\mathcal{O}$ is a *plain observation*, and is denoted by a list $\langle \ell_1, \ldots, \ell_n \rangle$ of *plain messages*.

Example 4 A fragmented observation relevant to viewer $\mathcal{V}_\psi$ is $\mathcal{O}_\psi = \langle \mu_1, \ldots, \mu_6 \rangle$, where $\mu_1 = (\{sh\}, \emptyset)$, $\mu_2 = (\{o_1\}, \{\mu_1\})$, $\mu_3 = (\{l, \varepsilon\}, \{\mu_1\})$, $\mu_4 = (\{o_2\}, \{\mu_2, \mu_3\})$, $\mu_5 = (\{c_2\}, \{\mu_4\})$, and $\mu_6 = (\{c_1, r\}, \{\mu_5\})$. The relevant observation graph $\gamma(\mathcal{O}_\psi)$ is depicted on the left of Fig. 4. □

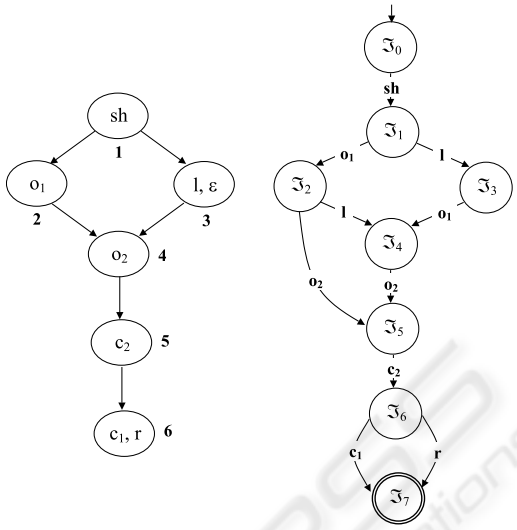

Figure 4: Observation graph (left) and relevant index space (right).

4.3 Index Space

Since it is neither trivial nor efficient to reason about the observation graph as is, an additional (acyclic) automaton is considered, called the *index space* of the observation,

$$\mathfrak{I}(\mathcal{O}) = (\mathbb{S}, \mathbb{E}, \mathbb{T}, S_0, \mathbb{S}_\mathrm{f}),$$

where $\mathbb{S}$ is the set of states, $\mathbb{E} = \mathbf{V} - \{\varepsilon\}$ the set of events, $\mathbb{T}$ the transition function, S_0 the initial state, and $\mathbb{S}_\mathrm{f}$ the set of final states. The peculiarity of an index space lies in that each path from the root to a final node, called a *temporal sequence*, represents a mode in which labels may be chosen in the observation graph without violating the constraints imposed by temporal and logical uncertainty. The whole set of such paths is the *extension* of $\mathfrak{I}(\mathcal{O})$, denoted $\|\mathfrak{I}(\mathcal{O})\|$. Note how each path within the extension is in fact a plain observation consistent with $\mathcal{O}$.

Example 5 Shown on the right of Fig. 4 is the index space $\mathfrak{I}(\mathcal{O}_\psi)$. The only final state is $\Im_7$. $\|\mathfrak{I}(\mathcal{O}_\psi)\|$ includes six plain observations. □

4.4 Ruler

A ruler establishes what transitions are faulty. Let $\mathbf{T}$ be the set of transitions relevant to components in Σ, and $\mathbf{R}$ a set of labels including the *null* label ε. A ruler $\mathcal{R}$ is a mapping from $\mathbf{T}$ to $\mathbf{R}$. If $(T, \varepsilon) \in \mathcal{R}$, then T is *normal*, otherwise T is *faulty*. A history segment h of Σ is said to *imply* a diagnosis δ, where δ is the set of faults associated with the faulty transitions of h defined in $\mathcal{R}$.

Example 6 A ruler $\mathcal{R}_\psi$ for Ψ can be defined by the set of faulty transitions, namely $\mathcal{R}_\psi = \{(T_1(p), s), (T_3(b_1), fo_1), (T_4(b_1), fc_1), (T_3(b_2), fo_2), (T_4(b_2, fc_2)\}$. □

5 DYNAMIC DIAGNOSIS

During its operation, a system Σ is expected to react to external events and to generate a collection of observable events that are received as a fragmented observation. Based on a fragmented problem $\wp(\Sigma) = (\Sigma_0, \mathcal{V}, \mathcal{O}, \mathcal{R})$, the goal of dynamic diagnosis is to compute the set of candidate diagnoses at the occurrence of each newly generated message.

At the beginning, no message is available, that is, the fragmented observation is empty, namely $\mathcal{O} = \langle\rangle$. However, in a strict sense, a first set of candidate diagnoses relevant to the empty observation should be provided, as Σ might have a silent reaction involving faulty transitions. At the occurrence of the first message μ_1, the monitoring is required to yield the set of candidate diagnoses relevant to $\mathcal{O}_{[1]} = \langle\mu_1\rangle$. More generally, if $\langle\mu_1, \ldots, \mu_k\rangle$ is the current sequence of messages, the occurrence of a new message μ_{k+1} causes the computation of the candidate diagnoses relevant to $\mathcal{O}_{[k+1]} = \langle\mu_1, \ldots, \mu_k, \mu_{k+1}\rangle$. Furthermore, for each message μ, dynamic diagnosis is required to compute the set of candidate diagnoses implied by the occurrence of μ, disregarding the whole set of candidate diagnoses relevant to the sequence of messages generated before μ. Thus, at the occurrence of the i-th message in $\mathcal{O}$, both such kinds of candidate sets are to be provided.

5.1 Silent Closure

Ideally, in order to perform dynamic diagnosis, we need to know the state reached by the system at the occurrence of each message. However, even in case the previous state is univocally known, the current state is bound to be uncertain owing to silent transitions and the uncertain nature of the message. On the other hand, the set of possible states at each newly generated message is confined within a limited domain, this corresponding to all the states reachable via silent transitions. This domain is a sort of (silent) closure of the current state, which encompasses the part of the behavior space that is transparent to the viewer. Formally, let σ_0 be a node of the behavior space $Bhv(\Sigma, \Sigma_0)$. The *silent closure*

$$Scl(\sigma_0) = (\mathbf{S}, \mathbf{E}, \mathbf{T}, S_0, \mathbf{S}_{\text{out}})$$

is an automaton such that $S_0 = (\sigma_0, \mathfrak{D}_0)$ is the *root*, and each state $S \in \mathbf{S}$ is a pair $(\sigma, \mathfrak{D})$ where σ is a state of $Bhv(\Sigma, \Sigma_0)$ and $\mathfrak{D}$ is a set of diagnoses δ where $\sigma_0 \rightsquigarrow \sigma$ is a history segment in $Bhv(\Sigma, \Sigma_0)$ that implies δ, called the *candidate attribute*. $\mathbf{E}$ is the set of transitions of Σ.

$\mathbf{T} : \mathbf{S} \times \mathbf{E} \mapsto \mathbf{S}$ is the transition function such that $(\sigma, \mathfrak{D}) \xrightarrow{T} (\sigma', \mathfrak{D}') \in \mathbf{T}$ if and only if T is a silent transition of Σ and $\sigma \xrightarrow{T} \sigma'$ is a transition in $Bhv(\Sigma, \Sigma_0)$.

$\mathbf{S}_{\text{out}} \subseteq \mathbf{S}$ is the *leaving set*, defined as follows. $S = (\sigma, \mathfrak{D}) \in \mathbf{S}_{\text{out}}$ if and only if there exists a visible transition $\sigma \xrightarrow{T} \sigma'$ in $Bhv(\Sigma, \Sigma_0)$.

Example 7 With reference to the behavior space outlined in Fig. 3, the silent closure of state 5, $Scl(5)$, is the subgraph involving states 5, 23, and 30, with $\mathbf{S}_{\text{out}} = \{23, 30\}$, whose states are left by visible transitions $T_2(b_2)$ and $T_4(p)$, respectively. □

5.2 Monitor

The *monitor* relevant to a system Σ, an initial state Σ_0, a viewer $\mathcal{V}$, and a ruler $\mathcal{R}$ is a graph

$$Mtr(\Sigma, \Sigma_0, \mathcal{V}, \mathcal{R}) = (\mathbb{N}, \mathbb{L}, \mathbb{E}, N_0),$$

where $\mathbb{N}$ is the set of nodes, $\mathbb{L}$ the set of labels, $\mathbb{E}$ the set of edges, and N_0 the initial node. Each node $N \in \mathbb{N}$ is an automaton

$$N = (\mathbf{S}, \mathbf{E}, \mathbf{T}, S_0, \mathbf{S}_{\text{out}}) = Scl(S_0),$$

where $S_0 \in Bhv(\Sigma, \Sigma_0)$. Let

$$\mathbb{S}_{\text{out}} = \bigcup_{N \in \mathbb{N}} \mathbf{S}_{\text{out}}(N), \qquad \mathbb{S}_0 = \bigcup_{N \in \mathbb{N}} \{S_0(N)\},$$

and $\mathbf{V}$ and $\mathbf{R}$ the domain of labels in $\mathcal{V}$ and $\mathcal{R}$, respectively. Each edge $E \in \mathbb{E}$ is marked by a label in $\mathbb{S}_{\text{out}} \times (\mathbf{V} - \{\varepsilon\}) \times \mathbf{R} \times \mathbb{S}_0$. An edge

$$N \xrightarrow{(S, \ell, \varphi, S')} N',$$

where $S = (\sigma, \mathfrak{D})$ and $S' = (\sigma', \mathfrak{D}')$ are internal nodes of N and N', respectively, is such that:

(1) $S' = S_0(N')$, namely, S' is the root of N';

(2) $\sigma \xrightarrow{T} \sigma'$ is a transition in $Bhv(\Sigma, \Sigma_0)$;

(3) ℓ is the (visible) label associated with T in $\mathcal{V}$;

(4) φ is the label (possibly ε) associated with T in $\mathcal{R}$.

The initial node N_0 is such that $S_0(N_0) = (\Sigma_0, \mathfrak{D}_0)$.

Let N be a node of $Mtr(\Sigma, \Sigma_0, \mathcal{V}, \mathcal{R})$. The *local candidate set* $\Delta^{\text{loc}}(N)$ of N is the union of the candidate attributes relevant to the internal states of N.

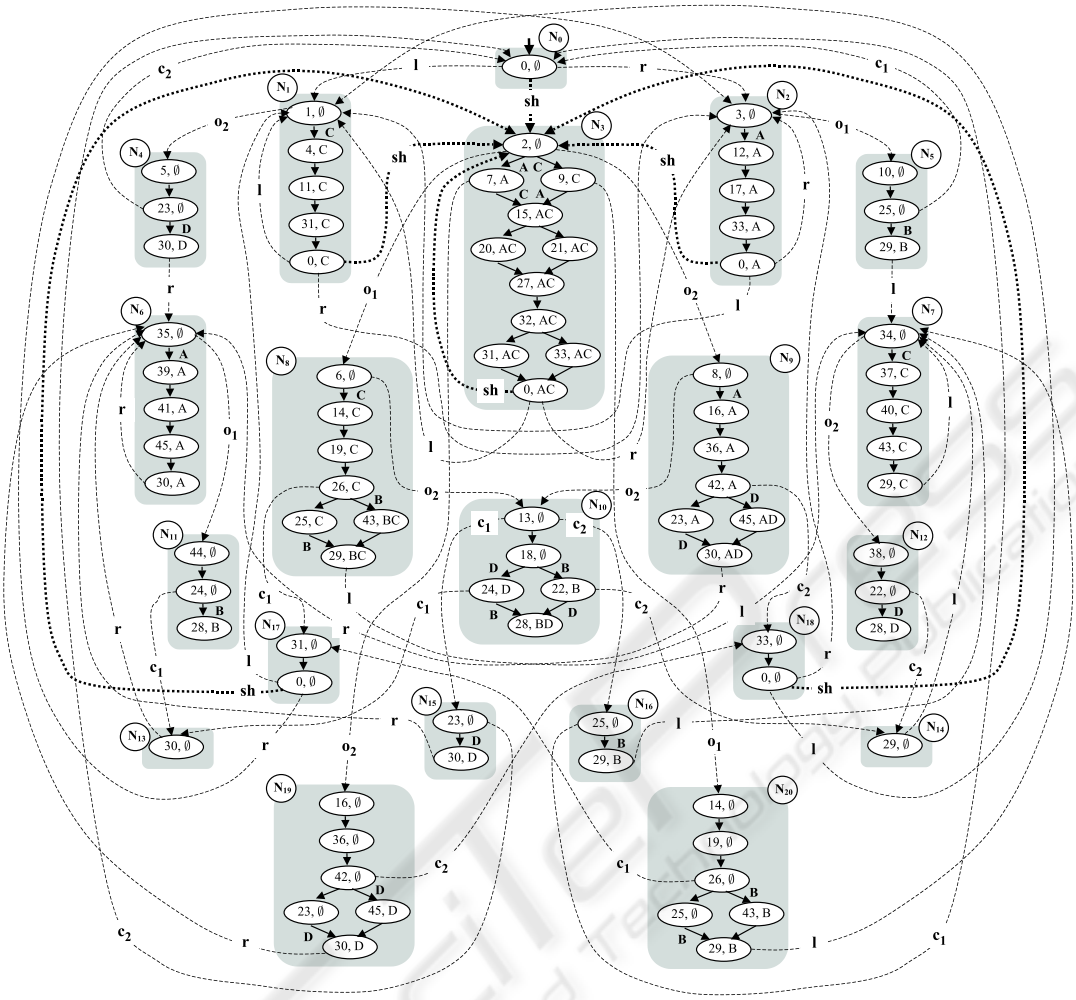

Figure 5: Monitor $Mtr(\Psi, \Psi_0, \mathcal{V}_\psi, \mathcal{R}_\psi)$.

Example 8 Portrayed in Fig. 5 is an abstract representation of the monitor $Mtr(\Psi, \Psi_0, \mathcal{V}_\psi, \mathcal{R}_\psi)$, where $\mathcal{V}_\psi$ and $\mathcal{R}_\psi$ are defined in Examples 3 and 6, respectively. Each node of the monitor is depicted within a shaded box and labeled by an identifier N_i, $i \in [0\,..\,20]$, where N_0 is the root. Within each node, faulty transitions are marked by letters A, B, C, or D, which are a shorthand for faults fo_1, fc_1, fo_2, and fc_2, respectively. Candidate attributes are written as strings of such letters, e.g., AC is a shorthand for $\{\{A, C\}\} = \{\{fo_1, fo_2\}\}$. Edges between nodes are represented as arrows from the internal state of the leaving node to the root of the entering node, and marked by the relevant viewer label. Identifiers of component transitions are omitted (see Fig. 3). □

5.3 Monitoring Trajectory

The notion of a monitor allows us to trace the state of the system based on a given fragmented observation. However, such a state is uncertain in nature for three reasons: (*i*) the uncertain nature of the message, (*ii*) the unobservability of the transitions within the nodes of the monitor, and (*iii*) the nondeterministic nature of the monitor, where different edges leaving the same node can be marked by the same observable label[4].

Let $\wp(\Sigma) = (\Sigma_0, \mathcal{V}, \mathcal{O}, \mathcal{R})$ be a fragmented diagnostic problem where $\mathcal{O}$ is a plain observation $\langle \ell_1, \ldots, \ell_n \rangle$, and $Mtr(\Sigma, \Sigma_0, \mathcal{V}, \mathcal{R}) = (\mathbb{N}, \mathbb{L}, \mathbb{E}, N_0)$ a relevant monitor. A *context* $\mathcal{N}$ is a triple $(N, \mathcal{S}, \mathcal{H})$, where N is a node in $\mathbb{N}$, $\mathcal{S}$ =

[4]For instance, considering Fig. 5, there are two different edges leaving node N_{10} marked by label c_1.

$\{\delta_1, \ldots, \delta_k\}$ is the *snapshot context*, where each δ is either an empty set or a singleton incorporating the label of a faulty transition, and $\mathcal{H} = \{\delta'_1, \ldots, \delta'_{k'}\}$ is the *historic context*, where each δ' is a set of labels relevant to faulty transitions.

The *diagnostic join* of two sets of diagnoses Δ_1 and Δ_2 is a set of diagnoses defined as follows:

$$\Delta_1 \bowtie \Delta_2 = \{\delta \mid \delta = \delta_1 \cup \delta_2, \delta_1 \in \Delta_1, \delta_2 \in \Delta_2\}.$$

A *monitoring state* $\mathcal{M}$ is a set $\{\mathcal{N}_1, \ldots, \mathcal{N}_m\}$ of contexts. The *monitoring trajectory* of $\wp(\Sigma)$, $Trj(\wp(\Sigma))$, is a sequence $\langle \mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_n \rangle$ of monitoring states (inductively) defined as follows:

(1) $\mathcal{M}_0 = \{(N_0, \{\emptyset\}, \{\emptyset\})\}$;

(2) For each $i \in [1\,..\,n]$, $\mathcal{M}_i$ is the minimal set of contexts $\mathcal{N}' = (N', \mathcal{S}', \mathcal{H}')$ such that $\mathcal{N} \in \mathcal{M}_{i-1}$, $\mathcal{N} = (N, \mathcal{S}, \mathcal{H})$, $S \in \mathbf{S}_{\text{out}}(N)$, $S = (\sigma, \mathcal{D})$, $N \xrightarrow{(S, \ell_i, \varphi, S_0(N'))} N' \in \mathbb{E}$, $\{\varphi\} \in \mathcal{S}'$, and $\mathcal{H}' \supseteq (\mathcal{D} \bowtie \mathcal{H} \bowtie \{\{\varphi\}\})$[5].

Example 9 Considering $Mtr(\Psi, \Psi_0, \mathcal{V}_\psi, \mathcal{R}_\psi)$ displayed in Fig. 5, assume the plain observation

$$\mathcal{O}'_\psi = \langle sh, o_1, l \rangle$$

and the corresponding diagnostic problem $\wp'(\Psi) = (\Psi_0, \mathcal{V}_\psi, \mathcal{O}'_\psi, \mathcal{R}_\psi)$. The trajectory $Trj(\wp'(\Psi))$ will be $\langle \mathcal{M}_0, \mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3 \rangle$, where

$\mathcal{M}_0 = \{(N_0, \{\emptyset\}, \{\emptyset\})\}$,
$\mathcal{M}_1 = \{(N_3, \{\{s\}\}, \{\{s\}\})\}$,
$\mathcal{M}_2 = \{(N_8, \{\emptyset\}, \{\{s\}\}), (N_{20}, \{\emptyset\}, \{\{s, C\}\})\}$,
$\mathcal{M}_3 = \{(N_7, \{\emptyset\}, \{\{s, B, C\}\})\}$. □

5.4 Candidate Sequence

Let $\wp(\Sigma)$ be a problem involving a plain observation. A *candidate pair* $\hat{\Delta}$ is an association $(\Delta^{\text{s}}, \Delta^{\text{h}})$ of two sets of diagnoses, where Δ^{s} is the *snapshot candidate set*, while Δ^{h} is the *historic candidate set*.

The *candidate sequence* $\hat{\boldsymbol{\Delta}}(\wp(\Sigma))$ is a list $\langle \hat{\Delta}_0, \hat{\Delta}_1, \ldots, \hat{\Delta}_n \rangle$ defined as follows. $\hat{\Delta}_0$ is the pair $(\Delta^{\text{loc}}(N_0), \Delta^{\text{loc}}(N_0))$, where N_0 is the root of $Mtr(\Sigma, \Sigma_0, \mathcal{V}, \mathcal{R})$, while for each $i \in [1\,..\,n]$, $\hat{\Delta}_i = (\Delta^{\text{s}}_i, \Delta^{\text{h}}_i)$, where:

$$\Delta^{\text{s}}_i = \bigcup_{(N, \mathcal{S}, \mathcal{H}) \in \mathcal{M}_i} (\Delta^{\text{loc}}(N) \bowtie \mathcal{S})$$

$$\Delta^{\text{h}}_i = \bigcup_{(N, \mathcal{S}, \mathcal{H}) \in \mathcal{M}_i} (\Delta^{\text{loc}}(N) \bowtie \mathcal{H})$$

where $\mathcal{M}_i$ is the monitoring state corresponding to the plain message ℓ_i in the trajectory $Trj(\wp(\Sigma))$.

[5] By definition, if $\varphi = \varepsilon$ then $\{\varphi\} = \emptyset$.

Example 10 With reference to the diagnostic problem $\wp'(\Psi)$ defined in Example 9, the candidate sequence $\hat{\boldsymbol{\Delta}}(\wp'(\Psi))$ will be $\langle \hat{\Delta}_0, \hat{\Delta}_1, \hat{\Delta}_2, \hat{\Delta}_3 \rangle$, where

$\hat{\Delta}_0 = (\{\emptyset\}, \{\emptyset\})$,
$\hat{\Delta}_1 = (\{\{s\}, \{s, A\}, \{s, C\}, \{s, A, C\}\},$
$\quad \{\{s\}, \{s, A\}, \{s, C\}, \{s, A, C\}\})$,
$\hat{\Delta}_2 = (\{\emptyset, \{B\}, \{C\}, \{B, C\}\},$
$\quad \{\{s\}, \{s, C\}, \{s, B, C\}\})$,
$\hat{\Delta}_3 = (\{\emptyset, \{C\}\}, \{\{s, B, C\}\})$.

Note how, at the arrival of the third message, the historic candidate set reduces from the three candidates of Δ^{h}_2 to the single diagnosis $\{s, B, C\}$ of Δ^{h}_3. □

5.5 Index-Space Decoration

The notions of monitoring trajectory and candidate sequence have been introduced based on plain observations. On the other hand, dynamic diagnosis is meant for solving diagnostic problems with fragmented observations, where messages are both logically and temporally uncertain. Such an observation is represented by a DAG from which an index space can be generated, as shown in Section 4.3. Each state of the index space corresponds to several possible ways in which observable labels may have been generated by the evolution of Σ, that is, to several plain observations. Thus, the computation of the candidate sequence, in the general case, requires associating each state $\Im$ of the index space with the set of monitoring states that are consistent with all the plain observations relevant to $\Im$. In other words, dynamic diagnosis requires an extension of the index space.

Let $\wp(\Sigma) = (\Sigma_0, \mathcal{V}, \mathcal{O}, \mathcal{R})$ be a fragmented diagnostic problem, and $\mathfrak{I}(\mathcal{O}) = (\mathbb{S}, \mathbb{E}, \mathbb{T}, S_0, \mathbb{S}_{\text{f}})$ the index space of $\mathcal{O}$. The *decoration* of $\mathfrak{I}(\mathcal{O})$ based on $\wp(\Sigma)$ is an automaton

$$\mathfrak{I}^*(\mathcal{O}) = (\mathbb{S}^*, \mathbb{E}^*, \mathbb{T}^*, S^*_0, \mathbb{S}^*_{\text{f}})$$

isomorphic to $\mathfrak{I}(\mathcal{O})$, where each state $S \in \mathbb{S}$ is marked by a *monitoring attribute* $\mathcal{M}$ as follows:

$$\mathcal{M} = \bigcup_{\mathcal{O}' \in \|S\|} \mathcal{M}_k$$

where $\|S\|$ denotes the set of plain observations up to S in $\mathfrak{I}(\mathcal{O})$, $\mathcal{O}' = \langle \ell_1, \ldots, \ell_k \rangle$, $\wp'(\Sigma) = (\Sigma_0, \mathcal{V}, \mathcal{O}', \mathcal{R})$, $Trj(\wp'(\Sigma)) = \langle \mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_k \rangle$.

Example 11 Consider the diagnostic problem $\wp(\Psi) = (\Psi_0, \mathcal{V}_\psi, \mathcal{O}_\psi, \mathcal{R}_\psi)$, where $\mathcal{V}_\psi$, $\mathcal{O}_\psi$, and

Table 1: Generation of the diagnostic sequence $\hat{\mathbf{\Delta}}(\wp(\Psi))$.

i	$\mathbb{S}^*_{\mathrm{f}_i}$	Δ^{s}_i	Δ^{h}_i
0	$\{\Im_0\}$	$\{\emptyset\}$	$\{\emptyset\}$
1	$\{\Im_1\}$	$\{\{s\},\{s,C\},\{s,A,C\}\}$	$\{\{s\},\{s,C\},\{s,A,C\}\}$
2	$\{\Im_2\}$	$\{\emptyset,\{B\},\{C\},\{B,C\}\}$	$\{\{s\},\{s,C\},\{s,B,C\}\}$
3	$\{\Im_4\}$	$\{\emptyset,\{C\}\}$	$\{\{s,B,C\}\}$
4	$\{\Im_5\}$	$\{\emptyset,\{B\},\{D\},\{B,D\}\}$	$\{\{s\},\{s,B\},\{s,D\},\{s,B,C\},\{s,B,D\},\{s,B,C,D\}\}$
5	$\{\Im_6\}$	$\{\emptyset,\{B\}\}$	$\{\{s\},\{s,B\},\{s,B,C\}\}$
6	$\{\Im_7\}$	$\{\emptyset\}$	$\{\{s\}\}$

$\mathcal{R}_\psi$ are defined in Examples 3, 4, and 6, respectively. Both the graph and the index space of $\mathcal{O}$ are depicted in Fig. 4. The decoration of $\mathfrak{I}(\mathcal{O})$ can be expressed by determining each monitoring attribute $\mathcal{M}_i$ that is relevant to node $\Im_i$, $i \in [0\,..\,7]$, namely:

$\mathcal{M}_0 = \{(N_0, \{\emptyset\}, \{\emptyset\})\}$,
$\mathcal{M}_1 = \{(N_3, \{\{s\}\}, \{\{s\}\})\}$,
$\mathcal{M}_2 = \{(N_8, \{\emptyset\}, \{\{s\}\}), (N_{20}, \{\emptyset\}, \{\{s, C\}\})\}$,
$\mathcal{M}_3 = \{(N_1, \{\emptyset\}, \{\{s, A, C\}\})\}$,
$\mathcal{M}_4 = \{(N_7, \{\emptyset\}, \{\{s, B, C\}\})\}$,
$\mathcal{M}_5 = \{(N_{10}, \{\emptyset\}, \{\{s\}\}), (N_{12}, \{\emptyset\}, \{\{s, B, C\}\})\}$,
$\mathcal{M}_6 = \{(N_{16}, \{\emptyset\}, \{\{s\}\}), (N_{14}, \{\emptyset\}, \{\{s, B\}\}),$
$(N_{14}, \{\emptyset\}, \{\{s, B, C\}\})\}$,
$\mathcal{M}_7 = \{(N_0, \{\emptyset\}, \{\{s\}\})\}$. □

Based on the concept of index-space decoration, both notions of monitoring trajectory and candidate sequence can be straightforwardly generalized to diagnostic problems involving a fragmented observation.

Let $\wp(\Sigma)$ be a problem involving a fragmented observation $\mathcal{O} = \langle \mu_1, \ldots, \mu_n \rangle$. Let $\mathfrak{I}^*(\mathcal{O}_{[i]}) = (\mathbb{S}^*_i, \mathbb{E}^*_i, \mathbb{T}^*_i, S^*_{0_i}, \mathbb{S}^*_{\mathrm{f}_i})$ be the decorated index space relevant to the sub-observation $\mathcal{O}_{[i]}$, $i \in [0\,..\,n]$. The trajectory of $\wp(\Sigma)$ is the sequence of monitoring states $\langle \mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_n \rangle$, where

$$\forall i \in [0\,..\,n] \left(\mathcal{M}_i = \bigcup_{(\Im, \mathcal{M}) \in \mathbb{S}^*_{\mathrm{f}_i}} \mathcal{M} \right).$$

The definition of the candidate sequence $\hat{\mathbf{\Delta}}(\wp(\Sigma))$ does not change, as $\mathcal{M}_0$ only depends on the root of the monitor, while each pair $\hat{\Delta}_i$, $i \in [1\,..\,n]$, depends upon the monitoring state $\mathcal{M}_i$ within the trajectory.

Example 12 With reference to the diagnostic problem $\wp(\Psi)$ defined in Example 11, the candidate sequence $\hat{\mathbf{\Delta}}(\wp(\Psi))$ will be $\langle \hat{\Delta}_0, \hat{\Delta}_1, \ldots, \hat{\Delta}_6 \rangle$, as detailed in Table 1. Specifically, each sub-observation $\mathcal{O}_{[i]}$ is associated with the set of final states $\mathbb{S}^*_{\mathrm{f}_i}$ of the decoration $\Im^*(\mathcal{O}_{[i]})$, whose monitoring attributes were computed in Example 11. Note how the historic candidate set reduces to a singleton $\{\{s\}\}$ upon the arrival of the sixth message. That is, a short circuit has occurred on the protected line and the protection apparatus has reacted correctly. □

Algorithm 1 The *Diagnose* procedure performs dynamic diagnosis by generating the candidate sequence relevant to a fragmented diagnostic problem $\wp(\Sigma) = (\Sigma_0, \mathcal{V}, \mathcal{O}, \mathcal{R})$. Each diagnostic pair within the sequence is generated at the arrival of each message of $\mathcal{O}$. Lines 1–3 create the roots of the monitor and of the decorated index space based on the empty observation. The core of the algorithm is the loop enclosed between Lines 5–17. Each iteration of the loop is triggered by the arrival of a new message (Line 6), which causes the extension of the decorated index space and of the monitoring space (Lines 7–9). At this point, possible monitoring states correspond to the monitoring attributes $\mathcal{M}$ of the current final states of the decorated index space. Both snapshot and historic candidate sets are computed based on the contexts incorporated in such attributes $\mathcal{M}$ (Lines 12–13), so that the corresponding candidate pair can be eventually generated (Line 16).

procedure *Diagnose*($\wp(\Sigma)$)
input
$\wp(\Sigma) = (\Sigma_0, \mathcal{V}, \mathcal{O}, \mathcal{R})$: a fragmented problem;
side effects
Incremental generation of $Mtr(\Sigma, \Sigma_0, \mathcal{V}, \mathcal{R})$,
Incremental generation of $\mathfrak{I}^*(\mathcal{O}) = (\mathbb{S}^*, \mathbb{E}^*, \mathbb{T}^*, S^*_0, \mathbb{S}^*_{\mathrm{f}})$,
Generation of a candidate pair at each new message;
begin
1. Create the root N_0 of $Mtr(\Sigma, \Sigma_0, \mathcal{V}, \mathcal{R})$;
2. Create the index space of the empty observation;
3. Mark the node of the index space with $\{(N_0, \{\emptyset\}, \{\emptyset\})\}$;
4. Generate $(\Delta^{\mathrm{loc}}(N_0), \Delta^{\mathrm{loc}}(N_0))$;
5. **loop**
6. $\mu :=$ the newly received message;
7. Extend the index space based on μ;
8. Extend the monitor based on Step 7;
9. Mark each new state of the index space

with the relevant monitoring attribute $\mathcal{M}$;
10. **for each** $(\Im, \mathcal{M}) \in \mathbb{S}_{\mathrm{f}}^{*}$ **do**
11. **for each** $(N, \mathcal{S}, \mathcal{H}) \in \mathcal{M}$ **do**
12. $\Delta^{\mathrm{s}} := \Delta^{\mathrm{s}} \cup (\Delta^{\mathrm{loc}}(N) \bowtie \mathcal{S})$;
13. $\Delta^{\mathrm{h}} := \Delta^{\mathrm{h}} \cup (\Delta^{\mathrm{loc}}(N) \bowtie \mathcal{H})$
14. **end-for**
15. **end-for**;
16. Generate $(\Delta^{\mathrm{s}}, \Delta^{\mathrm{h}})$
17. **end-loop**

end.

6 CONCLUSION

This paper has introduced a method for computing diagnoses during monitoring of a class of asynchronous DESs that keep on being called active systems although they have been endowed with the new feature of a (local and global) *priority hierarchy*. Also the notion of a diagnostic problem has changed, having been introduced both a *ruler* and a *viewer* in its definition. These new concepts enable to decouple the (behavioral) models of system components from the descriptions of their observability and abnormality properties.

In the literature only a more limited separation between component models and observability properties can be found: in (Console et al., 2002) each specific problem assigns the same observability to all the instances of the same component type whereas each instance can be endowed with distinct properties in the current approach. It is worth noting that (Console et al., 2002) gives the conceptual means to *characterize* model-based diagnosis, not to compute diagnoses, whereas our proposal encompasses also operational methods. As to the separation between component models and abnormality properties, this belongs exclusively to the approach described in the present paper.

The essential novelty of the paper, however, is the extension of dynamic diagnosis to *fragmented observations*, these being uncertain observations (Lamperti and Zanella, 2002) such that observable events are received one by one and the reception order does not reflect the emission order. Each received message consists of a logical content and a (possibly) empty temporal content. The logical content is uncertain in that it may range over a set of labels, each of which may have been emitted by several components. The available temporal content is uncertain since it does not allow, in general, to determine one emission order, instead, it is compliant with several ones. A limiting *monotonicity* assumption implicit in the notion of a fragmented observation is that the temporal content of each newly received message cannot place the emission of such a message after that of any message that has not been received yet.

The adopted algorithm for dynamic diagnosis adapts that described in (Lamperti and Zanella, 2003a). However, in (Lamperti and Zanella, 2003a) it was assumed that the label inherent to each received message was precisely identified and the reception order exactly matched the emission order. In the new algorithm, in order to handle fragmented observations, the observation index space is not computed beforehand as in (Lamperti and Zanella, 2002), instead, it is built incrementally, by updating it every time a new message is received. This incremental construction, which directly leads to a deterministic index space without any need of generating a nondeterministic one first, could indeed be proficiently exploited also by a posteriori diagnosis.

The dynamic diagnosis algorithm is inherently nonmonotonic since, as in (Lamperti and Zanella, 2003a), any estimate of the current system state may not survive a new message. Orthogonally, owing to temporal uncertainty in fragmented observations, every time a new message is received further sequences of labels may have to be added to the ones hypothesized so far. However, the monotonicity assumption prevents any sequence of labels hypothesized in previous monitoring steps to be refuted. Future research will tackle the relaxation of the monotonicity assumption, thus introducing a second source of nonmonotonicity to be coped with by the reasoning mechanism. Another plan for future work is to apply the modeling and reasoning principles described in this paper to a real-world apparatus.

In the literature, monitoring-based diagnosis of DESs is considered also by the diagnoser approach (Sampath et al., 1995; Sampath et al., 1996) and the incremental decentralized diagnoser approach (Pencolé et al., 2001). Both contributions differ from the current method in several aspects. First, the class of considered systems is different. In fact, while both the quoted approaches deal exclusively with synchronous DESs, the new method can cope with asynchronous ones, where every system may follow behavioral silent cycles over time (which is not the case for the diagnoser approach). Moreover, the extension of the current method to systems that integrate synchronous and asynchronous behavior is straightforward (they are already dealt with in (Lamperti and Zanella, 2003a), although considering certain plain observations only). Second, both approaches consider an observation without any uncertainty while the method introduced in this paper takes as input a fragmented observa-

tion. Major differences inherent to the adopted algorithms are highlighted and discussed in (Lamperti and Zanella, 2003a).

REFERENCES

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

Console, L., Picardi, C., and Ribaudo, M. (2002). Process algebras for systems diagnosis. *Artificial Intelligence*, 142(1):19–51.

Cordier, M. and Largouët, C. (2001). Using model-checking techniques for diagnosing discrete-event systems. In *Twelfth International Workshop on Principles of Diagnosis – DX'01*, pages 39–46, San Sicario, I.

Debouk, R., Lafortune, S., and Teneketzis, D. (2000). A diagnostic protocol for discrete-event systems with decentralized information. In *Eleventh International Workshop on Principles of Diagnosis – DX'00*, pages 41–48, Morelia, MX.

Fattah, Y. E. and Provan, G. (1997). Modeling temporal behavior in the model-based diagnosis of discrete-event systems (a preliminary note). In *Eighth International Workshop on Principles of Diagnosis – DX'97*, Mont St. Michel, F.

Lamperti, G. and Pogliano, P. (1997). Event-based reasoning for short circuit diagnosis in power transmission networks. In *Fifteenth International Joint Conference on Artificial Intelligence – IJCAI'97*, pages 446–451, Nagoya, J.

Lamperti, G. and Zanella, M. (2002). Diagnosis of discrete-event systems from uncertain temporal observations. *Artificial Intelligence*, 137(1–2):91–163.

Lamperti, G. and Zanella, M. (2003a). Continuous diagnosis of discrete-event systems. In *Fourteenth International Workshop on Principles of Diagnosis – DX'03*, pages 105–111, Washington DC.

Lamperti, G. and Zanella, M. (2003b). *Diagnosis of Active Systems – Principles and Techniques*, volume 741 of *The Kluwer International Series in Engineering and Computer Science*. Kluwer Academic Publisher, Dordrecht, NL.

Lunze, J. (2000). Diagnosis of quantized systems based on a timed discrete-event model. *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans*, 30(3):322–335.

Pencolé, Y., Cordier, M., and Rozé, L. (2001). Incremental decentralized diagnosis approach for the supervision of a telecommunication network. In *Twelfth International Workshop on Principles of Diagnosis – DX'01*, pages 151–158, San Sicario, I.

Sampath, M., Sengupta, R., Lafortune, S., Sinnamohideen, K., and Teneketzis, D. (1995). Diagnosability of discrete-event systems. *IEEE Transactions on Automatic Control*, 40(9):1555–1575.

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