INCREMENTAL PROCESSING OF TEMPORAL OBSERVATIONS IN SUPERVISION AND DIAGNOSIS OF DISCRETE-EVENT SYSTEMS

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: Diagnosis, supervision, discrete-event systems, temporal observations, indexing techniques, uncertainty.

Abstract: Observations play a major role in supervision and diagnosis of discrete-event systems (DESs). In a distributed, large-scale setting, the observation of a DES over a time interval is not perceived as a totally-ordered sequence of observable labels but, rather, as a directed acyclic graph, under uncertainty conditions. Problem solving, however, requires generating a surrogate of such a graph, the index space. Furthermore, the observation hypothesized so far has to be integrated at the reception of a new fragment of observation. This translates to the need for computing a new index space every time. Since such a computation is expensive, a naive generation of the index space from scratch at the occurrence of each observation fragment becomes prohibitive in real applications. To cope with this problem, the paper introduces an incremental technique for efficiently modeling and indexing temporal observations of DESs.

1 INTRODUCTION

Observations are the inputs to several tasks that can be carried out by exploiting Model-Based Reasoning techniques. Typically, they are the inputs to supervision, control and monitoring of physical processes (Rozé, 1997); they represent the symptoms of diagnosis (Brusoni et al., 1998); they are the clues for history reconstruction (Baroni et al., 1997; Baroni et al., 1999), and the test cases for software debugging (Wotawa, 202; Köb and Wotawa, 2004) . *Temporal observations* refer to dynamical systems and processes, and are endowed not only with a logical content, describing *what* has been observed, but also with a temporal content, describing *when* it has been observed.[1] Both (independent) aspects, can be modeled either quantitatively or qualitatively. This paper addresses the most qualitative abstraction of the notion of a temporal observation, i.e. an observation whose logical and temporal contents are both qualitative. This abstraction is quite important since adopting qualitative models is an issue of Model-Based Reasoning and Qualitative Physics as well, in far as reasoning about (a finite number of) qualitative values is easier and computationally cheaper. Moreover, adopting a higher abstraction level first, so as to focus attention, and a more detailed level later, is a principle of hierarchical model-based diagnosis (Mozetič, 1991). A general model for (qualitative uncertain) temporal observations was proposed in (Lamperti and Zanella, 2002), and exploited for describing the input of an a posteriori diagnosis task. Such a model consists of a directed acyclic graph where each node contains an uncertain logical content and each edge is a temporal precedence relationship. Thus, the graph, altogether, shows all the uncertain values observed over a time interval and their partial temporal ordering. Each uncertain logical content ranges over a set of qualitative values (labels). Therefore the observation graph implicitly represents all the possible sequences of labels consistent with the received temporal observation, where each sequence is a sentence of a language. Then, the observation graph, although intuitive and easy to build from the point of view of the observer, is unsuitable for processing. For any further processing it is better to represent a language in the standard way regular languages are represented (Aho et al., 1986), that is, by means of a deterministic automaton. In (Lamperti and Zanella, 2002) this automaton is called *index space* and it is built as the transformation of a nondeterministic automa-

[1] Indeed, in several contexts, the time a value is observed is not relevant, while it is interesting to know the time such a value was generated by the considered system/process.

Lamperti G. and Zanella M. (2006).
INCREMENTAL PROCESSING OF TEMPORAL OBSERVATIONS IN SUPERVISION AND DIAGNOSIS OF DISCRETE-EVENT SYSTEMS.
In *Proceedings of the Eighth International Conference on Enterprise Information Systems - AIDSS*, pages 47-57
DOI: 10.5220/0002447200470057

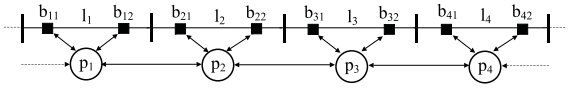

Figure 1: Power transmission network.

ton drawn from the observation graph. The problem arises when the nodes of the observation graph are received one at a time, typically in supervision and diagnosis of dynamical systems, in particular, discrete-event systems (DESs). In fact, the supervision process is required to react at each occurring piece of observation so as to generate appropriate diagnostic information (Lamperti and Zanella, 2004a; Lamperti and Zanella, 2004b). This translates to the need for generating a new index space at each new reception. However, a naive approach, that each time makes up the new index space from scratch, would be inadequate from the computational point of view. We need therefore an incremental technique for index-space generation.

2 APPLICATION DOMAIN

We consider a reference application domain of power networks. A power network is composed of transmission lines. Each line is protected by two breakers that are commanded by a protection. The protection is designed to detect the occurrence of a short circuit on the line based on the continuous measurement of its impedance: when the impedance goes beyond a given threshold, the two breakers are commanded to open, thereby causing the extinction of the short circuit. In a simplified view, the network is represented by a series of lines, each one associated with a protection, as displayed in Fig. 1, where lines $l_1 \cdots l_4$ are protected by protections $p_1 \cdots p_4$, respectively. For instance, p_2 controls l_2 by operating breakers b_{21} and b_{22}. In normal (correct) 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 latter is faulty, then p_2 will send a signal to p_3, which is supposed to command b_{32}. A recovery action may be faulty on its turn. For example, b_{32} may not open when tripped by p_2, thereby causing a further propagation of the recovery to protection p_4. The protection system is designed to propagate the recovery request until the tripped breaker opens correctly.

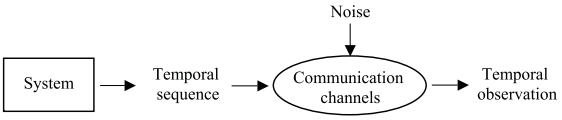

Figure 2: Genesis of a temporal observation.

When the protection system is reacting, a subset of the occurring events are visible to the operator in a control room who is in charge of monitoring the behavior of the network and, possibly, to issue explicit commands so as to minimize the extent of the isolated subnetwork. Typical visible events are *short* (a short circuit occurred on the line), *open* (a breaker opened), *close* (a breaker closed), and *end* (the short circuit extinguished). Generally speaking, however, 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 observation, it is impractical for the operator to reason on the observations so as to make consistent hypotheses on the behavior of the system and, eventually, to establish the shorted line and the faulty breakers.

3 TEMPORAL OBSERVATIONS

A temporal observation $\mathcal{O}$ is the mode in which the observable labels, generated by the evolution of a DES, are perceived by the observer. Considering the realm of asynchronous DESs, such as *active systems* (Lamperti and Zanella, 2003), a *history* h of a system is a sequence of component transitions, where each transition refers to a communicating automaton. Thus, $h = \langle T_1, \ldots, T_n \rangle$. Since a subset of the transitions are visible, the system history is expected to generate a sequence of observable labels, namely a *temporal sequence* $\langle \ell_1, \ldots, \ell_k \rangle$, where each ℓ_i, $i \in [1 \,..\, k]$, is the label generated by a visible transition in h. However, due to the multiplicity of communication channels between the (distributed) system and the observer, and to noise on such channels, the temporal observation $\mathcal{O}$ received by the observer is likely to differ from the temporal sequence generated by the system (see Fig. 2). Intuitively, $\mathcal{O}$ is a sequence of temporal fragments bringing information about what/when something is observed. Formally, let Λ be a domain of observable labels, possibly including the *null label* ε.[2] A *temporal fragment* φ is a

[2]The null label ε is invisible to the observer.

pair (λ, τ), where $\lambda \subseteq \Lambda$ is called the *logical content*, and τ is a set of fragments, called the *temporal content*. Specifically, $\mathcal{O}$ is a sequence of temporal fragments, $\mathcal{O} = \langle \varphi_1, \ldots, \varphi_n \rangle$, such that

$$\forall i \in [1\,..\,n], \varphi_i = (\lambda_i, \tau_i)\ \ (\tau_i \subseteq \{\varphi_1, \ldots, \varphi_{i-1}\})\,.$$

The temporal content of a fragment φ is supposed to refer to a (possibly empty) subset of the fragments preceding φ in $\mathcal{O}$. Thus, a fragment is uncertain in nature, both logically and temporally. Logical uncertainty means that λ includes the actual (possibly null) label generated by a system transition, but further spurious labels may be involved too. Temporal uncertainty means that only partial ordering is known among fragments. As such, both logical and temporal uncertainty are a sort of relaxation of the temporal sequence generated by the system, the former relaxing the actual visible label into a set of candidate labels, the latter relaxing absolute temporal ordering into partial ordering.

A *sub-observation* $\mathcal{O}_{[i]}$ of $\mathcal{O}$, $i \in [0\,..\,n]$, is the (possibly empty) prefix of $\mathcal{O}$ up to the i-th fragment, $\mathcal{O}_{[i]} = \langle \varphi_1, \ldots, \varphi_i \rangle$.

Example 1. Let $\Lambda = \{short, open, \varepsilon\}$, $\mathcal{O} = \langle \varphi_1, \varphi_2, \varphi_3, \varphi_4 \rangle$, where $\varphi_1 = (\{short, \varepsilon\}, \emptyset)$, $\varphi_2 = (\{open, \varepsilon\}, \{\varphi_1\})$, $\varphi_3 = (\{short, open\}, \{\varphi_2\})$, $\varphi_4 = (\{open\}, \{\varphi_1\})$. φ_1, is logically uncertain (either *short* or nothing is observed). φ_2 follows φ_1 and is logically uncertain ($open$ vs. nothing). φ_3 follows φ_2 and is logically uncertain (*short* vs. *open*). φ_4 follows φ_1 and is logically certain (*open*). However, no temporal relationship is defined between φ_4 and φ_2 or φ_3. □

Based on Λ, a temporal observation $\mathcal{O} = \langle \varphi_1, \ldots, \varphi_n \rangle$ can be represented by a DAG, called an *observation graph*,

$$\gamma(\mathcal{O}) = (\Lambda, \Omega, \mathbb{E})$$

where $\Omega = \{\omega_1, \ldots, \omega_n\}$ is the set of nodes isomorphic to the fragments in $\mathcal{O}$, each node being marked by a nonempty subset of Λ, and $\mathbb{E}$ is the set of edges isomorphic to the temporal content of fragments in $\mathcal{O}$:

$$\forall \omega_i \in \Omega, \varphi_i = (\lambda_i, \tau_i), \varphi_j \in \tau_i\ (N_j \mapsto N_i \in \mathbb{E}),$$
$$\forall (\omega_j \mapsto \omega_i \in \mathbb{E})\ (\varphi_j \in \tau_i, \varphi_i = (\lambda_i, \tau_i)).$$

A *precedence relationship* is defined between nodes of the graph, 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 $\omega = \omega'$.

Example 2. Consider the observation $\mathcal{O}$ defined in Example 1. The relevant observation graph $\gamma(\mathcal{O})$ is shown in Fig. 3. Note how $\gamma(\mathcal{O})$ implicitly contains several candidate temporal sequences, each candidate sequence generated by picking up a label from each node of the graph without violating the partially-ordered temporal relationships among nodes. Possible candidates are, among others, $\langle short, open, short, open \rangle$, $\langle short, open, open \rangle$, and $\langle short, open \rangle$.[3] However, we do not know which of the candidates is the actual temporal sequence generated by the system, the other ones being the *spurious* candidate sequences. Consequently, from the observer viewpoint, all candidate sequences share the same ontological status. □

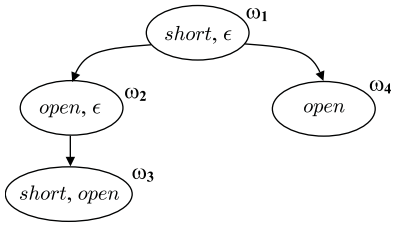

Figure 3: Observation graph $\gamma(\mathcal{O})$.

4 INDEXING OBSERVATIONS

The rationale of the paper is that, both for computational and space reasons, *the observation graph is inconvenient for carrying out a task that takes as input a temporal observation*. This claim applies to *linear observations* as well, which are merely a sequence O of observable labels. In this case, it is more appropriate to represent each sub-observation $O' \subseteq O$ as an integer index i corresponding to the length of O'. As such, i is a surrogate of O'. The same approach have been proposed for graph-based temporal observations (Lamperti and Zanella, 2000). However, we need extending the notion of an index appropriately and make model-based reasoning on a surrogate of the temporal observation, called an index space.

Let $\gamma(\mathcal{O}) = (\Lambda, \Omega, \mathbb{E})$ be an observation graph. A *prefix* $\mathcal{P}$ of $\mathcal{O}$ is a (possibly empty) subset of Ω where

$$\forall\, \omega \in \mathcal{P}\ (\nexists\, \omega' \in \mathcal{P}\ (\omega' \prec \omega)).$$

The formal definition of an index space is supported by the introduction of two functions on $\mathcal{P}$. The set of *consumed nodes* up to $\mathcal{P}$ is

$$Cons(\mathcal{P}) = \{\omega \mid \omega \in \Omega, \omega' \in \mathcal{P}, \omega \preceq \omega'\}. \quad (1)$$

The set of *consumable nodes* from $\mathcal{P}$, called the *frontier* of $\mathcal{P}$, is defined as

$$Front(\mathcal{P}) = \{\omega \mid \omega \in (\Omega - Cons(\mathcal{P})),$$
$$\forall (\omega' \mapsto \omega) \in \mathbb{E}\ (\omega' \in Cons(\mathcal{P}))\}. \quad (2)$$

[3] The fact that the length of a candidate temporal sequence may be shorter than the number of nodes in the observation graph comes from the immateriality of the null label ε, which is 'transparent'. For instance, candidate $\langle \varepsilon, \varepsilon, short, open \rangle$ is in fact $\langle short, open \rangle$.

Example 3. Considering $\gamma(\mathcal{O})$ in Fig. 3, with $\mathcal{P} = \{\omega_2, \omega_4\}$, we have $Cons(\mathcal{P}) = \{\omega_1, \omega_2, \omega_4\}$ and $Front(\mathcal{P}) = \{\omega_3\}$. □

The two functions defined on an index are formally related to one another by Theorem 1.

Theorem 1. *The frontier of an index $\Im$ is empty iff the set of consumable nodes of $\Im$ equals the set of nodes of the observation graph:*

$$Front(\Im) = \emptyset \iff Cons(\Im) = \Omega. \qquad (3)$$

Proof (sketch). When $\Omega = \emptyset$ (empty observation), Eq. (3) is trivially proven. Thus, we assume $\Omega \neq \emptyset$. Considering $Front(\Im) = \emptyset$, based on Eq. (2), we have $\{\omega \mid \omega \in (\Omega - Cons(\Im)), \forall(\omega' \mapsto \omega) \in \mathbb{E}\ (\omega' \in Cons(\Im))\} = \emptyset$. Assume $Cons(\Im) \subset \Omega$. Let $\bar{\omega} \in (\Omega - Cons(\Im))$. Three scenarios are possible:

(a) $\forall(\omega' \mapsto \bar{\omega}) \in \mathbb{E}\ (\omega' \in Cons(\Im))$. In this case, Eq. (2) is satisfied for $\omega = \bar{\omega}$, that is, $Front(\Im) \neq \emptyset$, a contradiction.

(b) $\nexists\omega'\ (\omega' \mapsto \bar{\omega} \in \mathbb{E})$. Even in this case, Eq. (2) is satisfied for $\omega = \bar{\omega}$, as the condition $\forall(\omega' \mapsto \omega) \in \mathbb{E}\ (\omega' \in Cons(\Im))$ is trivially true. Thus, $Front(\Im) \neq \emptyset$, a contradiction.

(c) $\exists(\omega' \mapsto \bar{\omega}) \in \mathbb{E}\ (\omega' \notin Cons(\Im))$. This scenario makes the universal predicate in Eq. (2) false. However, we may choose a different node in $(\Omega - Cons(\Im))$, rather than $\bar{\omega}$, specifically one of the ω' satisfying the existential predicate in scenario (c). This way, the above three scenarios are possible for the new node too. In particular, if either (a) or (b) holds, $Front(\Im) \neq \emptyset$, a contradiction. If (c) holds, the same considerations are repeated recursively, until either (a) or (b) holds. Since the observation graph is a DAG, such a recursion will end at the either (a) or (b), that is, $Front(\Im) \neq \emptyset$, a contradiction.

Thus, $Front(\Im) = \emptyset \implies Cons(\Im) = \Omega$. Conversely, assume $Cons(\Im) = \Omega$ and $Front(\Im) \neq \emptyset$. Based on Eq. (2), $Front(\Im)$ includes an ω in $(\Omega - Cons(\Im)) = \emptyset$, a contradiction. Thus, $Cons(\Im) = \Omega \implies Front(\Im) = \emptyset$. □

Let $\mathcal{O}$ be a temporal observation. The *prefix space* of $\mathcal{O}$ is the nondeterministic automaton

$$Psp(\mathcal{O}) = (\mathbb{S}^{\mathrm{n}}, \mathbb{L}^{\mathrm{n}}, \mathbb{T}^{\mathrm{n}}, S_0^{\mathrm{n}}, \mathbb{S}_{\mathrm{f}}^{\mathrm{n}})$$

where

$$\mathbb{S}^{\mathrm{n}} = \{\mathcal{P} \mid \mathcal{P} \text{ is a prefix of } \mathcal{O}\}$$

is the set of states,

$$\mathbb{L}^{\mathrm{n}} = \{\ell \mid \ell \in \lambda, (\lambda, \tau) \in \Omega\}$$

is the set of labels,

$$S_0^{\mathrm{n}} = \emptyset$$

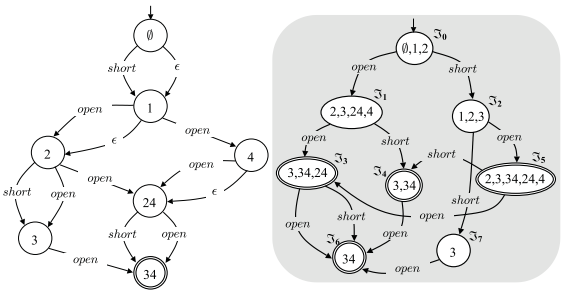

Figure 4: Prefix space $Psp(\mathcal{O})$ and index space $Isp(\mathcal{O})$, where $\mathcal{O}$ is depicted in Fig. 3.

is the initial state,

$$\mathbb{S}_{\mathrm{f}}^{\mathrm{n}} = \{\mathcal{P} \mid \mathcal{P} \in \mathbb{S}^{\mathrm{n}}, Cons(\mathcal{P}) = \Omega\}$$

is the set of final states, and $\mathbb{T}^{\mathrm{n}} : \mathbb{S}^{\mathrm{n}} \times \mathbb{L}^{\mathrm{n}} \mapsto 2^{\mathbb{S}^{\mathrm{n}}}$ is the transition function such that $\mathcal{P} \xrightarrow{\ell} \mathcal{P}' \in \mathbb{T}^{\mathrm{n}}$ iff, defining the '$\oplus$' operation as

$$\mathcal{P} \oplus \omega = (\mathcal{P} \cup \{\omega\}) - \{\omega' \mid \omega' \in \mathcal{P}, \omega' \prec \omega\}, \quad (4)$$

we have:

$$\omega \in Front(\mathcal{P}), \omega = (\lambda, \tau), \ell \in \lambda, \mathcal{P}' = \mathcal{P} \oplus \omega.$$

The *index space* of $\mathcal{O}$ is the deterministic automaton

$$Isp(\mathcal{O}) = (\mathbb{S}, \mathbb{L}, \mathbb{T}, S_0, \mathbb{S}_{\mathrm{f}})$$

equivalent to $Psp(\mathcal{O})$. Each state in $Isp(\mathcal{O})$ is an index of $\mathcal{O}$. The peculiarity of an index space is that each path from S_0 to a final state is a mode in which we may choose a label in each node of the observation graph $\gamma(\mathcal{O})$ based on the partial ordering imposed by $\gamma(\mathcal{O})$ (Lamperti and Zanella, 2002).

Example 4. Consider $\gamma(\mathcal{O})$ in Fig. 3. Shown in Fig. 4 are the prefix space $Psp(\mathcal{O})$ (left) and the index space $Isp(\mathcal{O})$ (shaded). Each prefix is written as a string of digits, e.g., 24 stands for $\mathcal{P} = \{\omega_2, \omega_4\}$. Final state are double circled. According to the standard algorithm that transforms a nondeterministic automaton to a deterministic one (Aho et al., 1986), each node of $Isp(\mathcal{O})$ is identified by a subset of the nodes of $Psp(\mathcal{O})$. Nodes in $Isp(\mathcal{O})$ have been named $\Im_0 \cdots \Im_7$. These are the indexes of $\mathcal{O}$. □

As for observations, we may define a restriction of the index space up to the i-th fragment as follows. Let $Isp(\mathcal{O}) = (\mathbb{S}, \mathbb{L}, \mathbb{T}, S_0, \mathbb{S}_{\mathrm{f}})$ be an index space, where $\gamma(\mathcal{O}) = (\Lambda, \Omega, \mathbb{E})$, $\Omega = \{\omega_1, \ldots, \omega_n\}$. Let S be a node in $\mathbb{S}$. The *sub-node* $S_{[i]}$ of S, $i \in [0\,..\,n]$, is

$$S_{[i]} = \begin{cases} S_0 & \text{if } i = 0 \\ \{\Im \mid \Im \in S, \forall\omega_j \in \Im\ (j \leq i)\} & \text{otherwise.} \end{cases} \quad (5)$$

The *sub-index space* $Isp_{[i]}$ of $\mathcal{O}$, $i \in [0\,..\,n]$, is an automaton

$$Isp_{[i]}(\mathcal{O}) = (\mathbb{S}', \mathbb{L}', \mathbb{T}', S_0, \mathbb{S}'_{\mathrm{f}})$$

where

$$\mathbb{S}' = \{S' \mid S \in \mathbb{S}, S' = S_{[i]}, S' \neq \emptyset\}$$
$$\mathbb{T}' = \{T' \mid T \in \mathbb{T}, T = S_1 \xrightarrow{\ell} S_2, T' = S'_1 \xrightarrow{\ell} S'_2, S'_1 = S_{1[i]}, S'_1 \neq \emptyset, S'_2 = S_{2[i]}, S'_2 \neq \emptyset\}$$
$$\mathbb{L}' = \{\ell \mid S'_1 \xrightarrow{\ell} S'_2 \in \mathbb{T}'\}$$
$$\mathbb{S}'_{\mathrm{f}} = \{S' \mid S' \in \mathbb{S}', \Im \in S', Cons(\Im) = \{\omega_1, \ldots, \omega_i\}\}.$$

The formal relationship between sub-observations and sub-index spaces is stated by Theorem 2.

Theorem 2. *The sub-index space of an observation equals the index space of the sub-observation,*

$$Isp_{[i]}(\mathcal{O}) = Isp(\mathcal{O}_{[i]}). \quad (6)$$

Proof (sketch). The proof is supported by two lemmas. Lemma 2.1 is based on the relationship between a nondeterministic automaton, A^{n}, and its equivalent deterministic one, A^{d}, based on the subset construction (Aho et al., 1986). $Closure(N^{\mathrm{n}})$ denotes the closure of node N^{n} in A^{n}. This is the set made up by N^{n} and all the nodes that are reachable from N^{n} via ε-transitions in A^{n}. Lemma 2.2 is grounded on the definition of $Psp(\mathcal{O})$ and, particularly, on Eq. (4).

Lemma 2.1. *Let $N_1 \xrightarrow{\ell} N_2$ be a transition in an index space $Isp(\mathcal{O})$. Then,*

(i) There exists $\bar{N}_1, \bar{N}_1 \subseteq N_1, \bar{N}_1 \neq \emptyset$, such that

$$\forall \Im \in \bar{N}_1 \ (\Im \xrightarrow{\ell} \Im' \in Psp(\mathcal{O}), \Im' \in N_2, Closure(\Im') \subseteq N_2);$$

(ii) All $\Im' \in N_2$ meet the condition stated in point (i).

Lemma 2.2. *Let $\Im \xrightarrow{\ell} \Im'$ be a transition in $Psp(\mathcal{O})$. Let $Max(\Im)$ denote the most recent fragment of $\Im$ in $\mathcal{O}$, $Max(\Im) = i \mid \omega_i \in \Im, \forall \omega_j \in \Im \ (j \leq i)$. Then,*

$$Max(\Im') \geq Max(\Im).$$

Theorem 2 can be proved by induction on the nodes of the two automata in Eq. (6). The basis states the equality of the initial states. Let S_0 and S'_0 be the initial states of $Isp_{[i]}(\mathcal{O})$ and $Isp(\mathcal{O}_{[i]})$, respectively. Based on the lemmas above and Eq. (5), an index $\Im \in S_0$ belongs to the closure of the root of $Psp(\mathcal{O})$ and is only composed of nodes ω_j such that $j \leq i$. Thus, $\Im \in S'_0$. Conversely, $\Im' \in S'_0$ belongs to the closure of the root of $Psp(\mathcal{O}_{[i]})$ and is only composed of nodes ω_j such that $j \leq i$, thereby $\Im' \in S_0$. In other terms, $S_0 = S'_0$. Now we show the equality of the transition functions.

Assume a transition $N_1 \xrightarrow{\ell} N_2 \in Isp_{[i]}(\mathcal{O})$, where N_1 is also a node in $Isp(\mathcal{O}_{[i]})$. We show that the same transition is in $Isp(\mathcal{O}_{[i]})$ too. Since N_1 and N_2 are the sub-nodes of two nodes in $Isp(\mathcal{O})$, namely N_1^+ and N_2^+, respectively, N_2 will not include the indexes $\Im''$ such that $\Im \xrightarrow{\ell} \Im' \in Psp(\mathcal{O})$, $\Im'' \in Closure(\Im')$, and either $\Im \in (N_1^+ - N_1)$ or $\omega_j \in \Im''$ where $j > i$. Let $\Im_2 \in N_2$. As such, $\Im_2$ only includes indexes ω_j where $j \leq i$. Thus, $\Im_2$ is reachable in $Psp(\mathcal{O})$ from an index in N_1 via a path of transitions, the first being marked by ℓ and the other ones by ε, each of them relevant to a node ω_j where $j \leq i$. Therefore, there exists a transition in $Isp(\mathcal{O}_{[i]})$ that is rooted in N_1 and is marked by ℓ, reaching a node N'_2 embodying $\Im_2$. We have to show that $N'_2 = N_2$. Assume $\Im'_2 \in N'_2$. As such, $\Im'_2 \in Closure(\Im'_1)$ where $\Im_1 \xrightarrow{\ell} \Im'_1 \in Psp(\mathcal{O}_{[i]})$, $\Im_1 \in N_1$, and each involved transition is relevant to a node ω_j such that $j \leq i$. Thus, the same sequence of consumptions of ω_j applies to a path in $Psp(\mathcal{O})$ that is rooted in $\Im_1 \in N_1$. Consequently, $\Im'_2 \in N_2$. In other terms, $N_2 = N'_2$, thereby $N_1 \xrightarrow{\ell} N_2 \in Isp(\mathcal{O}_{[i]})$.

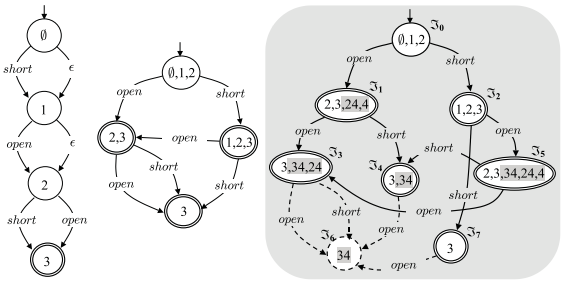

Figure 5: $Psp(\mathcal{O}_{[3]})$, $Isp(\mathcal{O}_{[3]})$, and $Isp_{[3]}(\mathcal{O})$.

Assume a transition $N_1 \xrightarrow{\ell} N'_2 \in Isp(\mathcal{O}_{[i]})$, where N_1 is also a node in $Isp_{[i]}(\mathcal{O})$. We show that the same transition is in $Isp_{[i]}(\mathcal{O})$. Consider an index $\Im'_2 \in N'_2$. As such, $\Im'_2 \in Closure(\Im'_1)$ where $\Im_1 \xrightarrow{\ell} \Im'_1 \in Psp(\mathcal{O}_{[i]})$, $\Im_1 \in N_1$, and each involved transition is relevant to a node ω_j such that $j \leq i$. Thus, there exists a transition in $Isp_{[i]}(\mathcal{O})$ that is rooted in N_1 and is marked by ℓ, reaching a node N_2 embodying $\Im'_2$. We have to show that $N_2 = N'_2$. Assume $\Im_2 \in N_2$. Following the same considerations applied above we come to the conclusion that $\Im_2$ is reachable in $Psp(\mathcal{O})$ from an index in N_1 via a path of transitions, the first being marked by ℓ and the other ones by ε, each of them relevant to a node ω_j where $j \leq i$. Therefore, there exists a transition in $Isp(\mathcal{O}_{[i]})$ that is rooted in N_1 and is marked by ℓ, reaching a node N'_2 embodying $\Im_2$. Therefore, $N_2 = N'_2$, hence, $N_1 \xrightarrow{\ell} N'_2 \in Isp_{[i]}(\mathcal{O})$, which concludes the induction step and the proof of the theorem. □

Example 5. Consider the observation $\mathcal{O}$ displayed in Fig. 3 and relevant index space in Fig. 4. We show

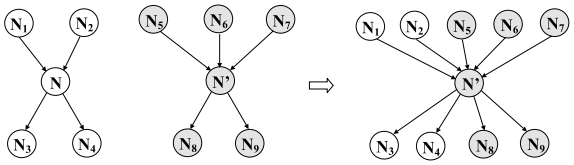

Figure 6: Effect of $Merge(N, N')$.

that, according to Theorem 2, $Isp_{[3]}(\mathcal{O}) = Isp(\mathcal{O}_{[3]})$. To this end, shown on the left-hand side of Fig. 5 is the prefix space $Psp(\mathcal{O}_{[3]})$, while the relevant index space $Isp(\mathcal{O}_{[3]})$ is depicted on the center. On the right-hand side of the figure is a transformation of the index space $Isp(\mathcal{O})$ outlined in Fig. 4. Specifically, each node S in $Isp(\mathcal{O})$ has been transformed into the subnode $S_{[3]}$ by removing some (possibly all) of the indexes, as established by Eq. (5). For instance, in node $\Im_5$, three, out of five indexes, have been dropped, namely 34, 24, and 4 (which stand for $\{\omega_3, \omega_4\}$, $\{\omega_2, \omega_4\}$, and $\{\omega_4\}$, respectively), thereby producing the sub-node marked by 2 and 3. Note how the sub-node of $\Im_6$ becomes empty after the removal of (the only) index 34. Based on the definition of sub-index space, empty nodes are not part of the result. This is why $\Im_6$ and all entering edges are in dotted lines. A further peculiarity is the occurrences of duplicated sub-nodes, as for example $\{\Im_3, \Im_4, \Im_7\}$ and $\{\Im_1, \Im_5\}$. Each set of replicated nodes forms an equivalence class of sub-nodes which results in fact in a single node in the sub-index space. Thus, $\{\Im_3, \Im_4, \Im_7\}$ and $\{\Im_1, \Im_5\}$ are collapsed into nodes 3 and 2, 3, respectively. This aggregation causes edges entering and/or exiting nodes in each equivalence class to be redirected to the corresponding subnode in the result. Performing such arrangements on the graph and removing the dotted part, we obtain in fact the same graph depicted on the center of Fig. 5, namely $Isp(\mathcal{O}_{[3]})$. This confirms Theorem 2. □

5 INCREMENTAL INDEXING

In case we need to perform the computation of the index space of each sub-observation of $\mathcal{O} = \langle \varphi_1, \ldots, \varphi_n \rangle$, namely $Isp(\mathcal{O}_{[i]})$, $i \in [1\,..\,n]$, the point is, *it is prohibitive to calculate each new index space from scratch at the occurrence of each fragment φ_i, as this implies the construction of the nondeterministic $Psp(\mathcal{O}_{[i]})$ and its transformation into the deterministic $Isp(\mathcal{O}_{[i]})$*. A better approach would be generating the new index space incrementally, based on the previous index space and the new observation fragment, avoiding the generation and transformation of the nondeterministic automaton $Psp(\mathcal{O})$.

This is performed by an algorithm called *Increment* (see below), which generates the new observation graph $\gamma(\mathcal{O}_{[i]})$ and relevant index space $Isp(\mathcal{O}_{[i]})$, based on the previous observation graph $\gamma(\mathcal{O}_{[i-1]})$, the relevant index space $Isp(\mathcal{O}_{[i-1]})$, and the new fragment φ_i. In so doing, *Increment* is supported by three auxiliary subroutines, *Clone*, *Merge*, and *Duplicate*. Such subroutines are defined in Lines 9–56, before the body of *Increment*.

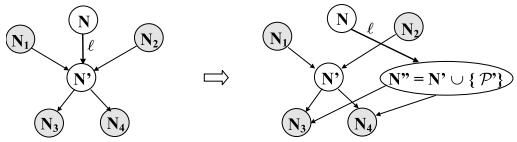

Figure 7: Effect of $Duplicate(N \xrightarrow{\ell} N', \mathcal{P}')$.

Function *Clone* (Lines 9–20) determines whether a node N^+, identified by the union of a set N of prefixes and a prefix $\mathcal{P}$ for $\mathcal{O}_{[i]}$, belongs to the current set $\mathbb{S}_i$ of nodes of the index space. If so, N^+ is returned, otherwise **nil** is returned.

Procedure *Merge* (Lines 21–38) merges two nodes N and N' of the index space, along with relevant edges, as shown in Fig. 6. This operation occurs when N is to be extended with a new prefix $\mathcal{P}'$, where $N \cup \{\mathcal{P}'\}$ is in fact N'. To do so, all edges entering/leaving N are redirected to/from N' (Lines 28–33), while N is removed (Line 34). In Lines 35–37, the set $\mathbb{B}$ is updated. $\mathbb{B}$ is a variable (initialized by *Increment*) called the *bud set*. Each element in $\mathbb{B}$ is a triple $(N, \mathcal{P}, \omega)$, called a *bud*, where N is a node of the index space, $\mathcal{P}$ a prefix in N, and ω a node of the observation graph belonging to the frontier of $\mathcal{P}$. A bud indicates that N needs further processing. Once processed, the bud is removed from $\mathbb{B}$. However, processing a bud possibly causes the generation of new buds. In Lines 35–37, *Merge* drops the buds relevant to N, while creating the corresponding buds in N'.

Procedure *Duplicate* (Lines 39–56) takes as input an edge $N \xrightarrow{\ell} N'$ of the index space and a prefix $\mathcal{P}'$. As shown in Fig. 7, it generates a new node $N^* = N' \cup \{\mathcal{P}'\}$ (Lines 46–47), redirects the input edge (Line 48), and duplicates all the edges leaving N' with corresponding edges leaving N^* (Lines 49–51). Finally, it updates the bud set by inserting new buds relevant to N^* (Line 52) and by duplicating the buds relevant to N' with corresponding buds in N^* (Lines 53–55).

The body of *Increment* is within Lines 57–114. It is conceptually divided into three sections: *initialization* (Lines 58–66), *core* (Lines 67–112), and *termination* (Line 113). The initialization section generates the new observation graph $\gamma(\mathcal{O}_{[i]})$ (Lines 58–61) and the initial values of the elements of $Isp(\mathcal{O}_{[i]})$ but $\mathbb{S}_{\mathrm{f}_i}$ (Lines 62–65). Besides, it creates the bud set $\mathbb{B}$, with initial buds based on the new fragment φ_i. The core

section is a loop iterating until the emptiness of the bud set. At each iteration, a new bud $B = (N, \mathcal{P}, \omega)$ is considered (Line 68) and a new prefix $\mathcal{P}'$ is generated (Line 69). Each label ℓ relevant to the logical content λ of ω is then considered (Lines 70–110). Eleven scenarios are to be distinguished, specifically:

(a) $\ell = \varepsilon$, $N' = N \cup \{\mathcal{P}'\}$ already exists in $\mathbb{S}_i$, and $N' \neq N$: N and N' are merged (Lines 71–75);

(b) $\ell = \varepsilon$, $N' = N \cup \{\mathcal{P}'\}$ already exists in $\mathbb{S}_i$, and $N' = N$: no operation is performed (Line 75);

(c) $\ell = \varepsilon$ and $N' = N \cup \{\mathcal{P}'\}$ is a new node: N is extended with $\mathcal{P}'$ and $\mathbb{B}$ is updated with the new buds relevant to $\mathcal{P}'$ (Lines 76–78);

(d) $\ell \neq \varepsilon$, there is no edge leaving N marked by ℓ, and $N' = \{\mathcal{P}'\}$ already exists: a new edge $N \xrightarrow{\ell} \{\mathcal{P}'\}$ is created (Lines 80–82);

(e) $\ell \neq \varepsilon$, there is no edge leaving N marked by ℓ, and $N' = \{\mathcal{P}'\}$ does not exists: a new node $N' = \{\mathcal{P}'\}$ and a new edge $N \xrightarrow{\ell} N'$ are created, and $\mathbb{B}$ is updated with the new buds relevant to $\mathcal{P}'$ (Lines 83–87);

(f) $\ell \neq \varepsilon$, there exists an edge $N \xrightarrow{\ell} N'$, no other edge enter N', $\bar{N} = N' \cup \{\mathcal{P}'\}$ already exists, and $\bar{N} \neq N'$: N' and $\bar{N}$ are merged (Lines 90–95);

(g) $\ell \neq \varepsilon$, there exists an edge $N \xrightarrow{\ell} N'$, no other edge enter N', $\bar{N} = N' \cup \{\mathcal{P}'\}$ already exists, and $\bar{N} = N'$: no operation is performed (Line 95);

(h) $\ell \neq \varepsilon$, there exists an edge $N \xrightarrow{\ell} N'$, no other edge enter N', and $\bar{N} = N' \cup \{\mathcal{P}'\}$ does not exist: $\mathcal{P}'$ is inserted into N' and $\mathbb{B}$ is updated with new buds relevant to $\mathcal{P}'$ (Lines 96–98);

(i) $\ell \neq \varepsilon$, there exists an edge $N \xrightarrow{\ell} N'$, there exists another edge entering N', $\bar{N} = N' \cup \{\mathcal{P}'\}$ already exists, and $\bar{N} \neq N'$: edge $N \xrightarrow{\ell} N'$ is substituted by $N \xrightarrow{\ell} \bar{N}$ (Lines 100–104);

(j) $\ell \neq \varepsilon$, there exists an edge $N \xrightarrow{\ell} N'$, there exists another edge entering N', $\bar{N} = N' \cup \{\mathcal{P}'\}$ already exists, and $\bar{N} = N'$: no operation is performed (Line 104);

(k) $\ell \neq \varepsilon$, there exists an edge $N \xrightarrow{\ell} N'$, there exists another edge entering N', and $N' \cup \{\mathcal{P}'\}$ does not exist: $Duplicate(N \xrightarrow{\ell} N', \mathcal{P}')$ is called (Lines 105–106).

In the termination section (Line 113), the set of final nodes of the new index space is generated: a node N is final iff it contains a prefix $\mathcal{P}$ whose frontier is empty.

1. $Increment(\gamma(\mathcal{O}_{[i-1]}), Isp(\mathcal{O}_{[i-1]}), \varphi_i, \gamma(\mathcal{O}_{[i]}), Isp(\mathcal{O}_{[i]}))$
2. **input**
3. $\gamma(\mathcal{O}_{[i-1]}) = (\Lambda_{i-1}, \Omega_{i-1}, \mathbb{E}_{i-1})$,
4. $Isp(\mathcal{O}_{[i-1]}) = (\mathbb{S}_{i-1}, \mathbb{L}_{i-1}, \mathbb{T}_{i-1}, S0_{i-1}, \mathbb{S}f_{i-1})$;
5. $\varphi_i = (\lambda_i, \tau_i)$: the i-th fragment of $\mathcal{O}$;
6. **output**
7. $\gamma(\mathcal{O}_{[i]}) = (\Lambda_i, \Omega_i, \mathbb{E}_i)$,
8. $Isp(\mathcal{O}_{[i]}) = (\mathbb{S}_i, \mathbb{L}_i, \mathbb{T}_i, S_{0_i}, \mathbb{S}_{f_i})$;

9. **function** $Clone(N, \mathcal{P})$: either a node in $\mathbb{S}_i$ or **nil**
10. **input**
11. $N = \{\mathcal{P}_1, \ldots, \mathcal{P}_k\}$: a set of prefixes of $\mathcal{O}_{[i]}$,
12. $\mathcal{P}$: a prefix of $\mathcal{O}_{[i]}$;
13. **begin** {Clone}
14. $N^+ := N \cup \{\mathcal{P}\}$;
15. **if** $N^+ \in \mathbb{S}_i$ **then**
16. **return**(N^+)
17. **else**
18. **return**(**nil**)
19. **end if**
20. **end** {Clone};

21. **procedure** $Merge(N, N')$
22. **input**
23. N: a node in $\mathbb{S}_i$,
24. N': a node in $\mathbb{S}_i$;
25. **side effects**
26. Merging of N and N';
27. **begin** {Merge}
28. **for each** $N'' \xrightarrow{\ell} N \in \mathbb{T}_i$ **do**
29. $\mathbb{T}_i := (\mathbb{T}_i \cup \{N'' \xrightarrow{\ell} N'\}) - \{N'' \xrightarrow{\ell} N\}$
30. **end for**;
31. **for each** $N \xrightarrow{\ell} N'' \in \mathbb{T}_i$ **do**
32. $\mathbb{T}_i := (\mathbb{T}_i \cup \{N' \xrightarrow{\ell} N''\}) - \{N \xrightarrow{\ell} N''\}$
33. **end for**;
34. $\mathbb{S}_i := \mathbb{S}_i - \{N\}$;
35. **for each** $(N, \mathcal{P}, \omega) \in \mathbb{B}$ **do**
36. $\mathbb{B} := (\mathbb{B} - \{(N, \mathcal{P}, \omega)\}) \cup \{(N', \mathcal{P}, \omega)\}$
37. **end for**
38. **end** {Merge};

39. **procedure** $Duplicate(N \xrightarrow{\ell} N', \mathcal{P}')$
40. **input**
41. $N \xrightarrow{\ell} N'$: an edge in $\mathbb{T}_i$,
42. $\mathcal{P}'$: a prefix of $\mathcal{O}_{[i]}$;
43. **side effects**
44. Creation of node $N^* = N' \cup \{\mathcal{P}'\}$ and edges;
45. **begin** {Duplicate}
46. $N^* := N' \cup \{\mathcal{P}'\}$;
47. $\mathbb{S}_i := \mathbb{S}_i \cup \{N^*\}$;
48. $\mathbb{T}_i := (\mathbb{T}_i - \{N \xrightarrow{\ell} N'\}) \cup \{N \xrightarrow{\ell} N^*\}$;
49. **for each** $N' \xrightarrow{x} N'' \in \mathbb{T}_i$ **do**
50. $\mathbb{T}_i := \mathbb{T}_i \cup \{N^* \xrightarrow{x} N''\}$
51. **end for**;
52. $\mathbb{B} := \mathbb{B} \cup \{(N^*, \mathcal{P}', \omega') \mid \omega' \in Front(\mathcal{P}')\}$;
53. **for each** $(N', \mathcal{P}, \omega) \in \mathbb{B}$ **do**
54. $\mathbb{B} := \mathbb{B} \cup \{(N^*, \mathcal{P}, \omega)\}$
55. **end for**
56. **end** {Duplicate};

57. **begin**{Increment}
58. $\Lambda_i := \Lambda_{i-1} \cup \lambda_i$;

59. ω_i := a new node marked by λ_i;
60. $\Omega_i := \Omega_{i-1} \cup \{\omega_i\}$;
61. $\mathbb{E}_i := \mathbb{E}_{i-1} \cup \{\omega \rightarrow \omega_i \mid \omega \in \tau_i\}$;
62. $\mathbb{S}_i := \mathbb{S}_{i-1}$;
63. $\mathbb{L}_i := \Lambda_i - \{\varepsilon\}$;
64. $\mathbb{T}_i := \mathbb{T}_{i-1}$;
65. $S_{0_i} := S_{0_{i-1}}$;
66. $\mathbb{B} := \{(N, \mathcal{P}, \omega_i) \mid N \in \mathbb{S}_i, \mathcal{P} \in N, \omega_i \in Front(\mathcal{P})\}$;
67. **repeat**
68. $B := (N, \mathcal{P}, \omega)$, where $B \in \mathbb{B}, \omega = (\lambda, \tau)$;
69. $\mathcal{P}' := \mathcal{P} \oplus \omega$;
70. **for each** $\ell \in \lambda$ **do**
71. **if** $\ell = \varepsilon$ **then**
72. **if** $(N' := Clone(N, \mathcal{P}')) \neq$ **nil then**
73. **if** $N' \neq N$ **then**
74. $Merge(N, N')$
75. **end if**
76. **else**
77. $N := N \cup \{\mathcal{P}'\}$;
78. $\mathbb{B} := \mathbb{B} \cup \{(N, \mathcal{P}', \omega') \mid \omega' \in Front(\mathcal{P}')\}$
79. **end if**
80. **elsif** $N \xrightarrow{\ell} N^* \notin \mathbb{T}_i$ **then**
81. **if** $\{\mathcal{P}'\} \in \mathbb{S}_i$ **then**
82. $\mathbb{T}_i := \mathbb{T}_i \cup \{N \xrightarrow{\ell} \{\mathcal{P}'\}\}$
83. **else**
84. $N' := \{\mathcal{P}'\}$;
85. $\mathbb{S}_i := \mathbb{S}_i \cup \{N'\}$;
86. $\mathbb{T}_i := \mathbb{T}_i \cup \{N \xrightarrow{\ell} N'\}$;
87. $\mathbb{B} := \mathbb{B} \cup \{(N', \mathcal{P}', \omega') \mid \omega' \in Front(\mathcal{P}')\}$
88. **end if**
89. **else**
90. N' := the node such that $T = N \xrightarrow{\ell} N' \in \mathbb{T}_i$;
91. **if** $\nexists T'(T' \in \mathbb{T}_i, T' = N'' \xrightarrow{\ell'} N', T' \neq T)$ **then**
92. **if** $(\bar{N} := Clone(N', \mathcal{P}')) \neq$ **nil then**
93. **if** $\bar{N} \neq N'$ **then**
94. $Merge(N', \bar{N})$
95. **end if**
96. **else**
97. $N' := N' \cup \{\mathcal{P}'\}$;
98. $\mathbb{B} := \mathbb{B} \cup \{(N', \mathcal{P}', \omega') \mid \omega' \in Front(\mathcal{P}')\}$
99. **end if**
100. **else**
101. **if** $(\bar{N} := Clone(N', \mathcal{P}')) \neq$ **nil then**
102. **if** $\bar{N} \neq N'$ **then**
103. $\mathbb{T}_i := (\mathbb{T}_i - \{N \xrightarrow{\ell} N'\}) \cup \{N \xrightarrow{\ell} \bar{N}\}$
104. **end if**
105. **else**
106. $Duplicate(N \xrightarrow{\ell} N', \mathcal{P}')$
107. **end if**
108. **end if**
109. **end if**
110. **end for**;
111. $\mathbb{B} := \mathbb{B} - \{B\}$
112. **until** $\mathbb{B} \neq \emptyset$;
113. $\mathbb{S}_{f_i} := \{N \mid N \in \mathbb{S}_i, \mathcal{P} \in N, Front(\mathcal{P}) = \emptyset\}$
114. **end** {Increment}.

Example 6. Consider the observation graph $\gamma(\mathcal{O})$ in Fig. 3, and the index space $Isp(\mathcal{O})$ shaded in Fig. 4. We now show the application of *Increment* that, based on $\gamma(\mathcal{O}_{[3]})$, $Isp(\mathcal{O}_{[3]})$, and φ_4, directly generates $Isp(\mathcal{O}_{[4]})$.

Shaded on the top-left of Fig. 8 is the computational state of procedure *Increment* after the *initialization* (ending at Line 66). The graph represents $Isp(\mathcal{O}_{[3]})$, with some extra information. Specifically, each bud $(N, \mathcal{P}, \omega_i) \in \mathbb{B}$ is represented by $\mathcal{P}^i$ in node N. For example, bud $(N_2, \{\omega_3\}, \omega_4)$ is written in N_2 as 3^4. The subsequent graphs in Fig. 8 depict the computational state of $Isp(\mathcal{O}_{[4]})$ at ech new iteration of the loop starting at Line 67. According to the initial (shaded) graph, at first, $\mathbb{B}$ includes eight buds.

Considering the *core* section, the loop (Lines 67–112) is iterated fourteen times:

(1) The chosen bud B in Line 68 is shaded in the corresponding pictorial representation. So, the bud picked up at the first iteration is $(N_3, \{\omega_3\}, \omega_4)$, where $\lambda(\omega_4) = \{open_1\}$. At Line 69, $\mathcal{P}' = \{\omega_3\} \oplus \omega_4 = \{\omega_3, \omega_4\}$. Consequently, the inner loop at Line 70 is iterated once for $\ell = open$. This corresponds to scenario (e) in the previous classification: the new node N_4 is created and linked from N_3 by an edge marked by $open$, as shown in graph $Step_1$. However, no new bud is inserted into $\mathbb{B}$, as $Front(\{\omega_3, \omega_4\}) = \emptyset$.

(2) $B = (N_2, \{\omega_3\}, \omega_4)$, $\lambda = \{open\}$, and $\mathcal{P}' = \{\omega_3, \omega_4\}$. This corresponds to scenario (k): node N_5 is generated by the *duplicate* procedure; however, no new bud is inserted into $\mathbb{B}$.

(3) $B = (N_1, \{\omega_3\}, \omega_4)$, $\lambda = \{open\}$, $\mathcal{P}' = \{\omega_2, \omega_4\}$, scenario (k): node N_6 is generated by duplication; moreover, a new bud $(N_6, \{\omega_2\}, \omega_4)$ is inserted.

(4) $B = (N_2, \{\omega_2\}, \omega_4)$, $\lambda = \{open\}$, $\mathcal{P}' = \{\omega_2, \omega_4\}$, scenario (k): node N_7 is generated by duplication; moreover, a new bud $(N_6, \{\omega_2, \omega_4\}, \omega_3)$ is created.

(5) $B = (N_7, \{\omega_2, \omega_4\}, \omega_3)$, $\lambda = \{short, open\}$, and $\mathcal{P}' = \{\omega_2, \omega_4\}$. For $\ell = short$, this corresponds to scenario (d): edge $N_7 \xrightarrow{open} N_4$ is created (Line 82). For $\ell = open$, scenario (j): no operation.

(6) $B = (N_6, \{\omega_2\}, \omega_4)$, $\lambda = \{open\}$, $\mathcal{P}' = \{\omega_2, \omega_4\}$, scenario (f): nodes N_5 and N_7 are merged.

(7) $B = (N_1, \{\omega_2\}, \omega_4)$, $\lambda = \{open\}$, $\mathcal{P}' = \{\omega_2, \omega_4\}$, scenario (h): node N_6 is extended with index $\mathcal{P}'$, and a new bud $(N_6, \{\omega_2, \omega_4\}, \omega_3)$ is created.

(8) $B = (N_6, \{\omega_2, \omega_4\}, \omega_3)$, $\lambda = \{short, open\}$, $\mathcal{P}' = \{\omega_3, \omega_4\}$. For $\ell = short$, scenario (k): node N_8 is generated by *duplicate*. For $\ell = open$, scenario (j): no operation.

(9) $B = (N_0, \{\omega_2\}, \omega_4)$, $\lambda = \{open\}$, $\mathcal{P}' = \{\omega_2, \omega_4\}$, and scenario (h): node N_2 is extended with index $\mathcal{P}'$, and a new bud $(N_2, \{\omega_2, \omega_4\}, \omega_3)$ is created.

(10) $B = (N_2, \{\omega_2, \omega_4\}, \omega_3)$, $\lambda = \{\lambda = \{short, open\}$, and $\mathcal{P}' = \{\omega_3, \omega_4\}$. For $\ell = short$,

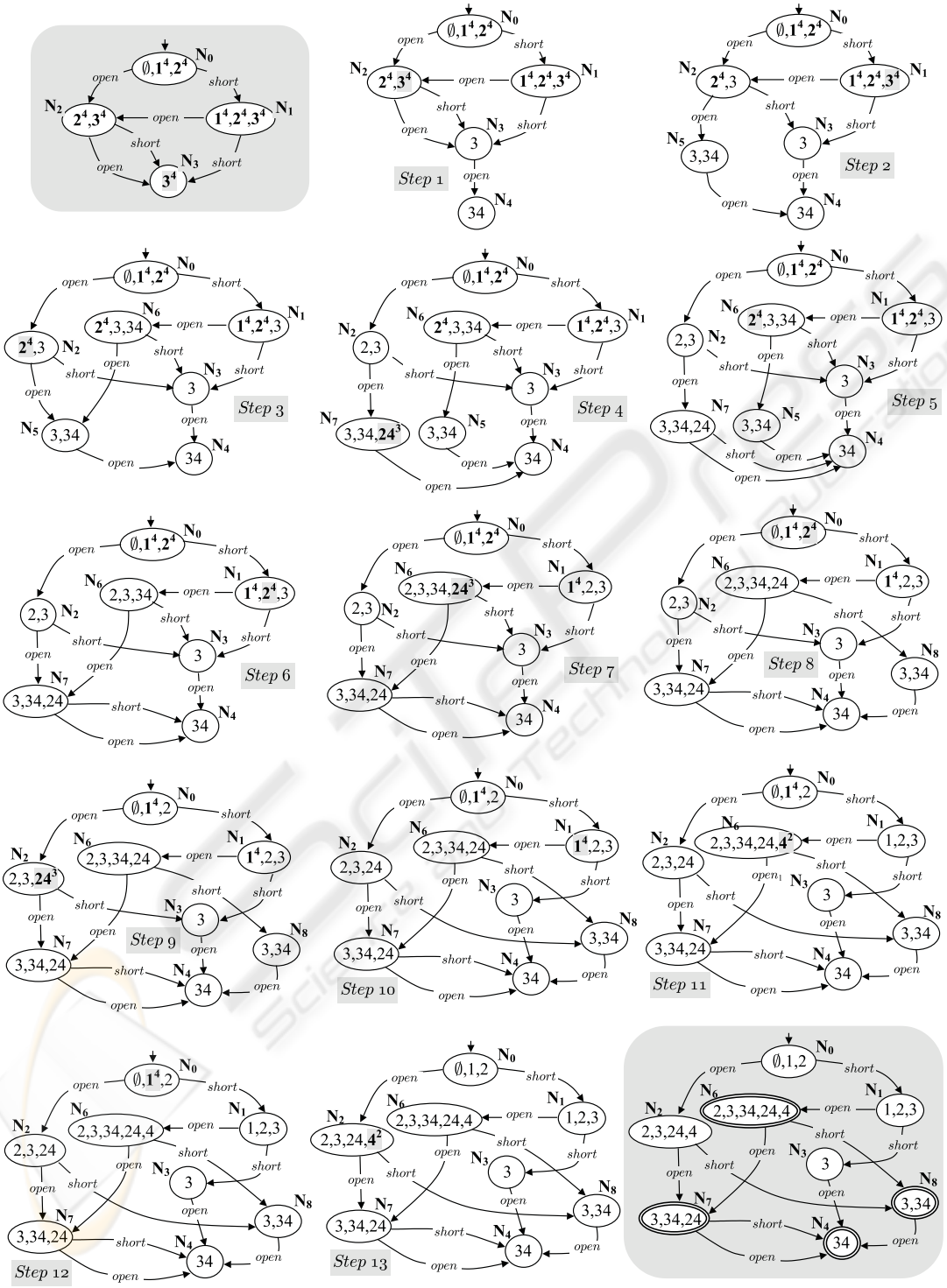

Figure 8: Tracing of the incremental computation of $Isp(\mathcal{O}_{[4]})$.

scenario (i): transition $N_2 \xrightarrow{short} N_3$ is redirected toward N_8. For $\ell = open$, scenario (j): no operation.

(11) $B = (N_1, \{\omega_1\}, \omega_4)$, $\lambda = \{open\}$, $\mathcal{P}' = \{\omega_4\}$, scenario (h): node N_6 is extended with $\mathcal{P}'$ and bud $(N_6, \{\omega_4\}, \omega_2)$ is created.

(12) $B = (N_6, \{\omega_4\}, \omega_2)$, $\lambda = \{open, \varepsilon\}$, $\mathcal{P}' = \{\omega_2, \omega_4\}$. For $\ell = open$, scenario (j): no operation. For $\ell = \varepsilon$, scenario (b): no operation.

(13) $B = (N_0, \{\omega_1\}, \omega_4)$, $\lambda = \{open\}$, $\mathcal{P}' = \{\omega_4\}$, scenario (h): node N_2 is extended with index $\mathcal{P}'$, and a new bud $(N_2, \{\omega_4\}, \omega_2)$ is created.

(14) $B = (N_2, \{\omega_4\}, \omega_2)$, $\lambda = \{open, \varepsilon\}$, $\mathcal{P}' = \{\omega_2, \omega_4\}$. For $\ell = open$, scenario (j): no operation. For $\ell = \varepsilon$, scenario (b): no operation.

Since the bud set $\mathbb{B}$ is empty, the *core* section ends. The *termination* section (Line 113) qualifies the final states of $Isp(\mathcal{O}_{[4]})$, namely N_4, N_6, N_7, and N_8. As expected, the last (shaded) graph in Fig. 8 represents the same automaton $Isp(\mathcal{O})$ in Fig. 4.

6 DISCUSSION

The technique for incremental construction of the index space is conceived in the context of dynamic model-based diagnosis of DESs. In this realm, the evolution of a system is monitored based on its model and the observation it generates during operation. The diagnostic engine is expected to react to each new fragment of observation by generating a corresponding set of *candidate diagnoses* based on the previous behavior of the system and the new fragment. As such, the diagnostic process is incremental in nature.

Model-based diagnosis of DESs is grounded on two essential elements: the observation $\mathcal{O}$ and the model $\mathfrak{M}$ of the system. Roughly, the diagnostic engine aims to explain $\mathcal{O}$ based on $\mathfrak{M}$. In so doing, a subset of the behavior space of the system is determined and represented by a finite automaton, where each path from the root to a final state is a candidate history of the system. Even if the automaton is finite, the number of candidate histories may be unbounded because of possible cycles within the automaton.

Each history is a sequence of component transitions, where each transition can be either normal or faulty. Consequently, each history corresponds to a candidate diagnosis, namely the set of faulty transitions within the history. Despite the unboundedness of candidate histories, the number of possible diagnoses is finite, since the total number of faulty transitions of components in the system is finite too. Precisely, if $\mathcal{F}$ is the domain of faulty transitions, the domain of candidate diagnoses is the powerset $2^{\mathcal{F}}$, including the empty diagnosis $\emptyset$ denoting a normal (rather than faulty) behavior. However, due to the constraints imposed by $\mathfrak{M}$ and $\mathcal{O}$, the set of candidate diagnoses is in general a small subset of $2^{\mathcal{F}}$, possibly a singleton.

Considering a diagnostic problem $\wp(\mathcal{O}, \mathfrak{M})$, where $\mathcal{O} = \langle \varphi_1, \ldots, \varphi_n \rangle$ is a temporal observation, we define the *static solution* of the problem, $\Delta(\wp(\mathcal{O}, \mathfrak{M}))$, the set of candidate diagnoses relevant to the histories drawn from $\mathcal{O}$ based on $\mathfrak{M}$. Since we are interested in updating the set of candidate diagnoses at each new fragment of observation, we have to consider the sequence of static solutions relevant to each *sub-problem* $\wp(\mathcal{O}_{[i]}, \mathfrak{M})$, $i \in [0\,..\,n]$, where $\mathcal{O}_{[i]}$ is the *sub-observation* $\langle \varphi_1, \ldots, \varphi_i \rangle$ up to the i-th fragment. Note that, when $i = 0$, we have an *empty* observation.

In other words, the diagnostic engine is expected to generate a new static solution at the occurrence of each newly-generated fragment of observation. This is called the *dynamic solution* of $\wp(\mathcal{O}, \mathfrak{M})$, namely

$$\mathbf{\Delta} = \langle \Delta(\wp(\mathcal{O}_{[0]}, \mathfrak{M})), \ldots, \Delta(\wp(\mathcal{O}_{[n]}, \mathfrak{M})) \rangle.$$

Another strong requirement for the diagnostic process is that the model $\mathfrak{M}$ of the system Σ is given only *intensionally*, that is, in terms of the topology of Σ (components and links among them) and the relevant component models (communicating finite automata), rather than *extensionally*, that is, in terms of the (possibly huge) automaton describing explicitly the system behavior.

On the other hand, just as the observation graph is not suitable for the diagnostic engine as is, and a surrogate of it (the index space) is used instead, the compositional model $\mathfrak{M}$ turns to be inadequate as is, and a surrogate of it is considered, namely the *model space* of $\mathfrak{M}$, denoted $Msp(\mathfrak{M})$. Thus, for computational reasons, the diagnostic problem $\wp(\mathcal{O}, \mathfrak{M})$ is transformed by the diagnostic engine into a surrogate $\wp(Isp(\mathcal{O}), Msp(\mathfrak{M}))$. As for the index space, the model space is made up incrementally, following a *lazy evaluation* approach: the model space is extended only when necessary for the diagnostic engine.

Essentially, a model space is a graph where nodes correspond to possible system states, while edges are marked by visible labels. Intuitively, a transition between nodes of the model space, $N \xrightarrow{\ell} N'$, occurs when the new observation fragment involves label ℓ.[4] Both nodes and edges of the model space carry compiled diagnostic information: the dynamic solution of the diagnostic problem can be generated based on such information provided the index space is somehow linked to the model space.

Specifically, each node $\Im$ of $Isp(\mathcal{O})$ must be *decorated* with the set of model-space states which comply

[4] Roughly, according to lazy evaluation, the generation of node N' is triggered by the occurrence of label ℓ.

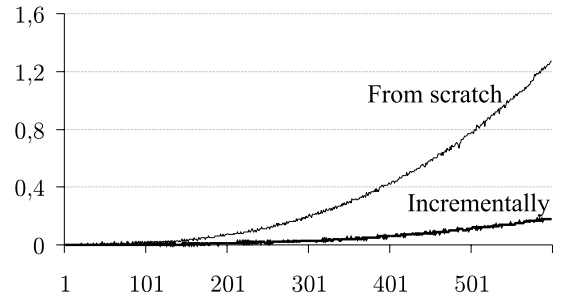

Figure 9: Experimental results: index-space computation-time (y-axis) vs. number of observation fragments (x-axis).

with all the paths up to $\Im$ in $Isp(\mathcal{O})$. Such a decoration is grounded on the common alphabet of the regular language of $Isp(\mathcal{O})$ and of $Msp(\mathfrak{M})$, namely the domain of visible labels. For example, if $\langle \ell_1, \ldots, \ell_k \rangle$ is a string of the language of $Isp(\mathcal{O})$ ending at node $\Im$, and the same sequence of labels is also a string of the language of $Msp(\mathfrak{M})$ ending at node N, then the decoration of $\Im$ will include N. Since several different strings may end at $\Im$, the decoration of $\Im$ will include several nodes of $Msp(\mathfrak{M})$. Accordingly, the *Increment* algorithm has been extended to cope with decorated index spaces too.

7 CONCLUSION

Both the observation graph and the index space are modeling primitives for representing temporal observations. Whereas the observation graph is the front-end representation, suitable for modeling an observation while it is being received over a time interval, the index space is a back-end representation, suitable for model-based problem-solving and as a standard interchange format of uncertain observations among distinct application contexts. This paper has presented a technique for constructing the index space incrementally, while receiving observation fragments one at a time. This is significant whenever a nonmonotonic processing step has to be performed after each observation fragment is received, as is when the tasks of supervision and dynamic diagnosis (and state estimation, in general), are considered. The *Increment* algorithm is an attempt to achieve the stated goals. Experimental results, shown in Fig. 9, indicate that the algorithm (implemented in C language) is sound, complete, and efficient. The diagram shows the time (in seconds) to compute the index space of an observation composed of (up to) 600 fragments. The curve on the top is relevant to the computation of each index space from scratch. The curve on the bottom corresponds to the incremental computation. The research still needs to perform computational analysis, and to gather further experimental results based on observations with different sizes and uncertainty degrees.

REFERENCES

Aho, A., Sethi, R., and Ullman, J. (1986). *Compilers – Principles, Techniques, and Tools*. Addison-Wesley, Reading, MA.

Baroni, P., Canzi, U., and Guida, G. (1997). Fault diagnosis through history reconstruction: an application to power transmission networks. *Expert Systems with Applications*, 12(1):37–52.

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

Brusoni, V., Console, L., Terenziani, P., and Dupré, D. T. (1998). A spectrum of definitions for temporal model-based diagnosis. *Artificial Intelligence*, 102(1):39–80.

Köb, D. and Wotawa, F. (2004). Introducing alias information into model-based debugging. In *Fifteenth International Workshop on Principles of Diagnosis – DX'04*, pages 93–98, Carcassonne, F.

Lamperti, G. and Zanella, M. (2000). Uncertain temporal observations in diagnosis. In *Fourteenth European Conference on Artificial Intelligence – ECAI'2000*, pages 151–155, Berlin, D.

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

Lamperti, G. and Zanella, M. (2004a). A bridged diagnostic method for the monitoring of polymorphic discrete-event systems. *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics*, 34(5):2222–2244.

Lamperti, G. and Zanella, M. (2004b). Dynamic diagnosis of active systems with fragmented observations. In *Sixth International Conference on Enterprise Information Systems – ICEIS'2004*, pages 249–261, Porto, P.

Mozetič, I. (1991). Hierarchical model-based diagnosis. *International Journal of Man-Machine Studies*, 35(3):329–362.

Rozé, L. (1997). Supervision of telecommunication network: a diagnoser approach. In *Eighth International Workshop on Principles of Diagnosis – DX'97*, pages 103–111, Mont St. Michel, F.

Wotawa, F. (202). On the relationship between model-based debugging and program slicing. *Artificial Intelligence*, 135(1–2):125–143.