ON CHECKING TEMPORAL-OBSERVATION SUBSUMPTION IN SIMILARITY-BASED DIAGNOSIS OF ACTIVE SYSTEMS

Gianfranco Lamperti, Federica Vivenzi and Marina Zanella
Dipartimento di Elettronica per l'Automazione, Via Branze 38, 25123 Brescia, Italy

Keywords: Similarity-based diagnosis, discrete-event systems, temporal observations, subsumption.

Abstract: Similarity-based diagnosis of large active systems is supported by reuse of knowledge generated for solving previous diagnostic problems. Such knowledge is cumulatively stored in a knowledge-base, when the diagnostic session is over. When a new diagnostic problem is to be faced, the knowledge-base is queried in order to possibly find a similar, reusable problem. Checking problem-similarity requires, among other constraints, that the observation relevant to the new problem be subsumed by the observation relevant to the problem in the knowledge-base. However, checking observation-subsumption, following its formal definition, is time and space consuming. The bottleneck lies in the generation of a nondeterministic automaton, its subsequent transformation into a deterministic one (the index space of the observation), and a regular-language containment-checking. In order to speed up the diagnostic process, an alternative technique is proposed, based on the notion of coverage. Besides being effective, subsumption-checking via coverage is also efficient because no index-space generation or comparison is required. Experimental evidence supports this claim.

1 INTRODUCTION

Discrete-event systems (DESs) (Cassandras and Lafortune, 1999) are dynamic systems, typically modeled as networks of components. Each component is a communicating automaton (Brand and Zafiropulo, 1983) that reacts to input events by state-transitions which possibly generate new events towards other components. Diagnosis of DESs is a challenging task that has been tackled since a decade via different approaches, either based on artificial intelligence (Pencolé, 2000; Rozé and Cordier, 2002; Console et al., 2002; Pencolé and Cordier, 2005) or automatic control techniques (Sampath et al., 1995; Sampath et al., 1996; Chen and Provan, 1997; Sampath et al., 1998; Zad et al., 1999; Cassandras and Lafortune, 1999; Lunze, 2000; Debouk et al., 2000; Schullerus and Krebs, 2001). Within the domain of a class of asynchronous DESs (Baroni et al., 1999; Lamperti and Zanella, 2003; Lamperti and Zanella, 2004; Lamperti and Zanella, 2006b), called *active systems*, a diagnosis approach has been proposed that is based on similarity techniques (Lamperti and Zanella, 2006a; Cerutti et al., 2007) with the aim of pursuing reuse of knowledge when solving a diagnostic problem. The idea is to store into a knowledge-base the data structures generated for solving each diagnostic problem. When a new problem is to be faced, instead of solving it from scratch, the knowledge-base is first browsed in order to find a previously-solved diagnostic problem that is 'compatible' with the new one. If so, the knowledge relevant to the old problem can be exploited to solve the new problem, thereby speeding up the diagnostic process. Among other constraints, such compatibility requires that the observation relevant to the problem in the knowledge-base subsume the observation relevant to the new problem. Such an observation is temporal in nature, and is represented by a DAG. The problem lies on the mode in which subsumption is checked, which, according to the definition of subsumption, is based on a containment relationship between the regular languages of the *index spaces* of the observations. The index space is a deterministic automaton whose generation (and comparison) may require considerable computational resources. So, this paper proposes an alternative, more efficient, approach for checking observation-subsumption that avoids index-space manipulation, by reasoning on the specific properties of the observations.

Lamperti G., Vivenzi F. and Zanella M. (2008).
ON CHECKING TEMPORAL-OBSERVATION SUBSUMPTION IN SIMILARITY-BASED DIAGNOSIS OF ACTIVE SYSTEMS.
In *Proceedings of the Tenth International Conference on Enterprise Information Systems - AIDSS*, pages 44-53
DOI: 10.5220/0001696200440053
Copyright © SciTePress

2 BACKGROUND

When an active system reacts, it generates a sequence of observable labels, called the *signature* of the reaction. However, what is actually perceived by the external observer is a *relaxation* of the signature $\mathbb{S}$. Such a relaxation is called a *temporal observation*. Formally, let $\mathcal{L}$ be the finite domain of all the observable labels the active system can generate, possibly including the *null* label ε. A temporal observation is a (not necessarily connected) DAG

$$\mathcal{O} = (\mathcal{N}, \mathcal{L}, \mathcal{A}) \tag{1}$$

where $\mathcal{N}$ is the set of nodes, with each $N \in \mathcal{N}$ being marked with a non-empty subset of $\mathcal{L}$, and $\mathcal{A} : \mathcal{N} \mapsto 2^{\mathcal{N}}$ is the set of arcs. A '$\prec$' temporal precedence relationship among nodes of the graph is defined as follows:

- If $N \mapsto N' \in \mathcal{A}$ then $N \prec N'$;
- If $N \prec N'$ and $N' \prec N''$ then $N \prec N''$;
- If $N \mapsto N' \in \mathcal{A}$ then $\nexists N'' \in \mathcal{N}\ (N \prec N'' \prec N')$.

The set of labels marking a node N is the *extension* of N, written $\|N\|$. Thus, the relaxation of the signature $\mathbb{S}$ into $\mathcal{O}$ involves three kinds of uncertainty:

- *Logical uncertainty*: each single observable label in the signature $\mathbb{S}$ is instead perceived as a set of candidate labels, possibly including the null label ε. All labels in $\|N\|$ but one are *spurious*, with just one being the *actual label*.[1]
- *Temporal uncertainty*: the absolute temporal ordering of the signature $\mathbb{S}$ is relaxed to partial temporal ordering. If $N \prec N'$ in $\mathcal{O}$, where ℓ and ℓ' are the actual labels in N and N', respectively, then ℓ precedes ℓ' in $\mathbb{S}$. However, not all precedence relationships between nodes in $\mathcal{N}$ are known.
- *Node uncertainty*: additional *spurious nodes* that involve ε (among other labels), are possibly inserted[2].

As such, $\mathcal{O}$ implicitly incorporates several *candidate signatures*, where each candidate is determined by selecting one label from each node in $\mathcal{N}$ without violating the temporal constraints imposed by the precedence relationships. The set of all the candidate signatures of $\mathcal{O}$ is called the *extension* of $\mathcal{O}$, written $\|\mathcal{O}\|$. Among such candidates is the actual signature $\mathbb{S}$. Like for nodes, all candidate signatures but one are spurious. The mode in which the signature $\mathbb{S}$ demeans to observation $\mathcal{O}$ is assumed to be unknown.[3] As explained i,n (Lamperti and Zanella, 2002), such a degradation may be caused by the multiplicity of the communication channels that convey observable labels from the system to the observer (temporal uncertainty), and to noise (logical uncertainty). However, although unknown, $\mathbb{S}$ is assumed to be preserved within $\mathcal{O}$.

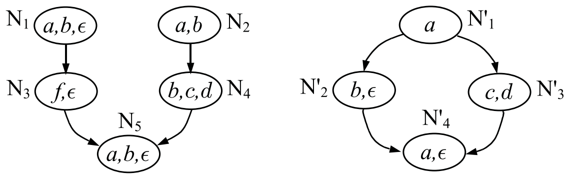

Figure 1: Observations $\mathcal{O}_1$ (left) and $\mathcal{O}_2$ (right).

Example 1. Shown in Fig. 1 are the graphs of two (both logically and temporally uncertain) observations, namely, from left to right, $\mathcal{O}_1 = (\mathcal{N}_1, \mathcal{L}_1, \mathcal{A}_1)$ and $\mathcal{O}_2 = (\mathcal{N}_2, \mathcal{L}_2, \mathcal{A}_2)$, where $\mathcal{N}_1 = \{N_1, \ldots, N_5\}$, $\mathcal{N}_2 = \{N'_1, \ldots, N'_4\}$, $\mathcal{L}_1 = \{a, b, c, d, f, \varepsilon\}$, and $\mathcal{L}_2 = \{a, b, c, d, \varepsilon\}$. In $\mathcal{O}_2$, N'_1 incorporates the first observable label, namely a. Then, either N'_2 or N'_3 follows, each of which involves two candidate labels, where ε is the null label. The last generated node is N'_4, with a and ε being the final candidate labels. The extension of the observation, namely $\|\mathcal{O}_2\|$, includes the candidate signatures *ac*, *ad*, *abc*, *abd*, *aca*, *ada*, *acb*, *adb*, *abca*, *abda*, *acba*, *adba*, each of which is obtained by selecting one label for each node without violating the temporal constraints, where the null label ε has been removed.

Within the diagnostic process it is inconvenient to reason on the observation $\mathcal{O}$ as is, mostly because the explanation-oriented diagnostic reasoning requires some sort of observation-indexing. Such an indexing is more naturally performed based on a surrogate of the observation, called the *index space*, namely $Isp(\mathcal{O})$. This is a deterministic automaton with the property that its regular language is the extension of $\mathcal{O}$,

$$Lang(Isp(\mathcal{O})) = \|\mathcal{O}\|. \tag{2}$$

In other words, the set of strings generated by each path in $Isp(\mathcal{O})$, from the initial state to a final state, equals the set of candidate signatures relevant to $\mathcal{O}$. As detailed in (Cerutti et al., 2007), the generation of the index space of $\mathcal{O}$ requires two steps, namely:

- Yielding the nondeterministic automaton, called the *prefix space* of $\mathcal{O}$, where each node identifies the set of consumed nodes in $\mathcal{N}$ up to now;

[1] If the actual label is ε, it means that no label was actually generated by the system. Note how the extension of a node in $\mathcal{N}$ cannot be the singleton $\{\varepsilon\}$.

[2] In a spurious node, the actual label is ε, with all the other labels being spurious.

[3] Otherwise, in principle, we might distill $\mathbb{S}$ from $\mathcal{O}$, thereby disregarding $\mathcal{O}$ in the diagnostic process.

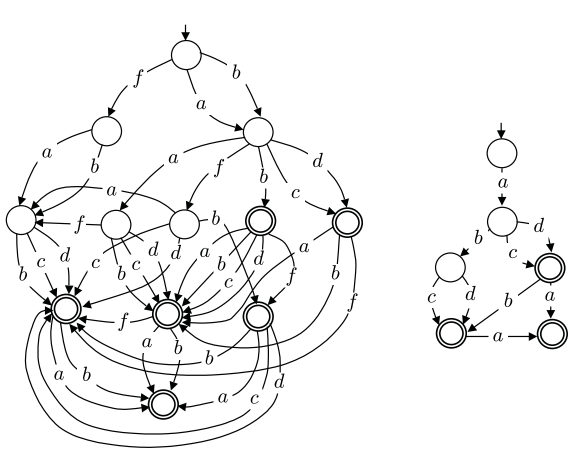

Figure 2: Index spaces $Isp(\mathcal{O}_1)$ (left) and $Isp(\mathcal{O}_2)$ (right) .

- Generating the deterministic automaton equivalent to the prefix space, in fact the index space.[4]

Furthermore, as explained shortly, the role of the index space comes into view for checking observation-subsumption too.

Example 2. Shown in Fig. 2 are the index spaces of observations $\mathcal{O}_1$ (left) and $\mathcal{O}_2$ (right) displayed in Fig. 1. It is easy to check that the regular language of each index space equals the extension of the relevant observation (the set of candidate signatures), where each string of the language corresponds to a path in the index space, from the initial state to one of the final states (with the latter being double circled in the figure). In particular, Example 1 offers evidence that $Lang(Isp(\mathcal{O}_2)) = \|\mathcal{O}_2\|$.

In similarity-based diagnosis of DESs (Lamperti and Zanella, 2006a), it is essential to understand whether the solution of the diagnostic problem $\wp'$ at hand can be supported by the knowledge yielded for solving a previous (different) diagnostic problem $\wp$, with the latter being stored in a knowledge-base. Among other constraints, reuse of $\wp$ can be exploited only if the observations $\mathcal{O}'$ and $\mathcal{O}$ relevant to $\wp'$ and $\wp$, respectively, are linked by a subsumption relationship,

$$\mathcal{O} \Supset \mathcal{O}' \tag{3}$$

namely, only if $\mathcal{O}$ *subsumes* $\mathcal{O}'$. The subsumption relationship is defined in terms of regular-language containment, relevant to the corresponding index spaces, precisely:

$$Lang(Isp(\mathcal{O})) \supseteq Lang(Isp(\mathcal{O}')). \tag{4}$$

[4]Being equivalent, both the prefix space and the index space share the same regular language (Hopcroft et al., 2006).

This means that $\mathcal{O}$ subsumes $\mathcal{O}'$ iff the set of candidate signatures of $\mathcal{O}$ includes all the candidate signatures of $\mathcal{O}'$.

The reason why observation subsumption supports reuse can be roughly explained as follows. The solution of $\wp$ yields an automaton μ, a sort of diagnoser, where each state is marked by a set of diagnoses and each transition is marked by a label in $\mathcal{L}$. The language of μ is the subset of the signatures relevant to $\mathcal{O}$ that comply with the model of the system, namely, $Lang(\mu) \subseteq \|\mathcal{O}\|$. The same applies to a new problem $\wp'$ relevant to $\mathcal{O}'$. However, if $\mathcal{O} \Supset \mathcal{O}'$, that is, $\|\mathcal{O}\| \supseteq \|\mathcal{O}'\|$, then $Lang(\mu) \supseteq Lang(\mu')$. In other words, μ contains all the signatures of μ'. This allows the diagnostic engine to reuse μ in order to generate μ' based on $\mathcal{O}'$. The advantage stems from the fact that such an operation is far more efficient than generating μ' from scratch, which would require heavy model-based reasoning.

Example 3. With reference to observations $\mathcal{O}_1$ and $\mathcal{O}_2$ outlined in Fig. 1, and the relevant index spaces in Fig. 2, it is easy to check that $\mathcal{O}_1 \Supset \mathcal{O}_2$, that is, $\|\mathcal{O}_1\| \supseteq \|\mathcal{O}_2\|$. In other words, each string in $Lang(Isp(\mathcal{O}_2))$ is also a string in $Lang(Isp(\mathcal{O}_1))$.

The problem with observation-subsumption checking lies on the fact that, establishing whether we can exploit the knowledge for solving $\wp$, in order to solve $\wp'$, requires a considerable amount of computational resources. Specifically, we need first generate $Isp(\mathcal{O}')$ and, subsequently, compare $Lang(Isp(\mathcal{O}'))$ with each $Lang(Isp(\mathcal{O}))$ in the knowledge-base, in the hope of finding a subsuming observation $\mathcal{O}$. Such an approach, based on the generation of the index space and on regular-language containment-checking, may be prohibitive in real applications. In order to cope with this complexity, we need some alternative checking-techniques.

3 CHECKING SUBSUMPTION

The systematic nature of checking based on the formal definition of subsumption stems primarily on its lack of prospection (short-sightedness). As a matter of fact, such a *systematic* technique does not perform any kind of reasoning on the given observations. Assume the problem of testing $\mathcal{O} \Supset \mathcal{O}'$, namely the *checking problem*. The idea is to find out some conditions that either imply or are implied by such a relationship. If these conditions can be checked using a reasonable amount of computational (space and time) resources, then chances are that we can give an answer to the checking problem efficiently. Specifically, if a necessary condition N_c is violated, then the answer to the checking problem will be *no*. Dually, if a sufficient condition S_c holds, then the answer will be *yes*. However, if either N_c holds or S_c is violated, then the checking problem remains unanswered. Necessary conditions and sufficient conditions relevant to the checking problem are given in Theorem 1 and Theorem 2, respectively. As shown shortly, these conditions are eventually incorporated within Algorithm 1 (see below).

Theorem 1. *Let $\mathcal{O} = (\mathcal{N}, \mathcal{L}, \mathcal{A})$ and $\mathcal{O}' = (\mathcal{N}', \mathcal{L}', \mathcal{A}')$ be two temporal observations. Let n and n' be the number of nodes in $\mathcal{N}$ and $\mathcal{N}'$, respectively. Let n_ε and n'_ε be the number of nodes that include the null label ε in $\mathcal{N}$ and $\mathcal{N}'$, respectively. Let $\mathcal{M}$ and $\mathcal{M}'$ be the multisets of observable labels occurring in $\mathcal{O}$ and $\mathcal{O}'$, respectively. Then, $\mathcal{O}$ subsumes $\mathcal{O}'$ only if the following conditions hold:*

$$n \geq n' \tag{5}$$

$$n_\varepsilon - n'_\varepsilon \geq n - n' \tag{6}$$

$$\mathcal{M} \supseteq \mathcal{M}'. \tag{7}$$

Proof. The proof is by contradiction. To prove condition (5), we have to show that $\mathcal{O} \Supset \mathcal{O}' \Rightarrow n \geq n'$. Assume the contrary, namely $n' > n$. Since $\forall N' \in \mathcal{N}'$ $(\|N'\| \neq \{\varepsilon\})$, we can make up a temporal sequence $\mathbb{T}$ by selecting a label $\ell \neq \varepsilon$ for each $N' \in \mathcal{N}'$, where $|\mathbb{T}| = n'$. Clearly, $\mathbb{T} \notin \|Isp(\mathcal{O})\|$ because temporal sequences relevant to $\mathcal{O}$ are long at most n. Hence, $\|Isp(\mathcal{O})\| \not\supseteq \|Isp(\mathcal{O}')\|$, that is, $\mathcal{O} \not\Supset \mathcal{O}'$, a contradiction.

To prove condition (6), we have to show that $\mathcal{O} \Supset \mathcal{O}' \Rightarrow n_\varepsilon - n'_\varepsilon \geq n - n'$. Assume the contrary, namely $n - n' > n_\varepsilon - n'_\varepsilon$ or, in other terms,

$$n - n_\varepsilon > n' - n'_\varepsilon. \tag{8}$$

Let $\mathcal{N}'_\varepsilon = \{N' \mid N' \in \mathcal{N}', \varepsilon \in \|N'\|\}$. Now, consider a sequence $\mathbb{L}'$ of labels selected from all nodes of $\mathcal{N}'$, in such a way that ε is chosen for all nodes in $\mathcal{N}'_\varepsilon$. Let $\mathbb{T}'$ be the temporal sequence corresponding to $\mathbb{L}'$. Clearly, $|\mathbb{T}'| = n' - n'_\varepsilon$. In consequence, $\mathbb{T}' \notin \|Isp(\mathcal{O})\|$ because each temporal sequence $\mathbb{T}$ relevant to $\mathcal{O}$ is such that $|\mathbb{T}| \geq n - n_\varepsilon$, that is, based on equation (8), $|\mathbb{T}| > n' - n'_\varepsilon$, hence, $|\mathbb{T}| > |\mathbb{T}'|$. Thus, $\|Isp(\mathcal{O})\| \not\supseteq \|Isp(\mathcal{O}')\|$, that is, $\mathcal{O} \not\Supset \mathcal{O}'$, a contradiction.

To prove condition (7), we have to show that $\mathcal{O} \Supset \mathcal{O}' \Rightarrow \mathcal{M} \supseteq \mathcal{M}'$. Assume the contrary, namely $\mathcal{M} \not\supseteq \mathcal{M}'$, that is, $\mathcal{M}' \supset \mathcal{M}$. This means that $\mathcal{M}'$ will contain $k' \geq 1$ occurrences of a label ℓ, with $\mathcal{M}$ including $k \geq 0$ occurrences of the same label, where $k' > k$. Choose $\mathbb{L}'$ so as to include all k' occurrences of ℓ in $\mathcal{O}'$. Hence, $\mathbb{T}'$ will contain exactly k' occurrences of ℓ. On the other hand, no temporal sequence $\mathbb{L}$ can be composed in $\mathcal{O}$ to include the same number of occurrences of ℓ. Thus, $\|Isp(\mathcal{O})\| \not\supseteq \|Isp(\mathcal{O}')\|$, that is, $\mathcal{O} \not\Supset \mathcal{O}'$, a contradiction. $\square$

Corollary 1.1. *$\mathcal{O}$ subsumes $\mathcal{O}'$ only if*

$$\mathcal{L} \supseteq \mathcal{L}'. \tag{9}$$

Proof. Condition (9) is entailed by condition (7) of Theorem 1, with the latter being necessary for $\mathcal{O} \Supset \mathcal{O}'$ to hold. Hence, condition (9) too is a necessary condition for observation subsumption. $\square$

Example 4. In Example 3 we have shown that $\mathcal{O}_1$ subsumes $\mathcal{O}_2$, where such observations are displayed in Fig. 1. Hence, the conditions relevant to Theorem 1 are expected to hold for $\mathcal{O}_1$ and $\mathcal{O}_2$. We have $n_1 = 5$, $n_2 = 4$, $n_{1\varepsilon} = 3$, $n_{2\varepsilon} = 2$. As a matter of fact, both conditions (5) and (6) hold. Moreover, since $\mathcal{M}_1 = [a,a,a,b,b,b,b,c,d,f,\varepsilon,\varepsilon,\varepsilon]$ and $\mathcal{M}_2 = [a,a,b,c,d,\varepsilon,\varepsilon]$, condition (7) holds too.

The conditions necessary for subsumption stated in Theorem 1 can be easily checked. Thus, they correspond to the first actions of the checking algorithm. If one of them is violated, the check terminates immediately with a negative answer. Otherwise, the check continues by testing a sufficient condition of subsumption based on the notion of coverage given in Definition 1 below. Roughly, $\mathcal{O}$ covers $\mathcal{O}'$ when $\mathcal{O}$ is a relaxation of $\mathcal{O}'$, inasmuch as an observation is a relaxation of a system signature.

Definition 1. (Coverage) *Let $\mathcal{O} = (\mathcal{N}, \mathcal{L}, \mathcal{A})$ and $\mathcal{O}' = (\mathcal{N}', \mathcal{L}', \mathcal{A}')$ be two temporal observations, where $\mathcal{N} = \{N_1, \ldots, N_n\}$ and $\mathcal{N}' = \{N'_1, \ldots, N'_{n'}\}$. We say that $\mathcal{O}$ covers $\mathcal{O}'$, written $\mathcal{O} \trianglerighteq \mathcal{O}'$, iff there exists a subset $\bar{\mathcal{N}}$ of $\mathcal{N}$, with $\bar{\mathcal{N}} = \{\bar{N}_1, \ldots, \bar{N}_{n'}\}$ having the same cardinality as $\mathcal{N}'$, such that, denoting $\mathcal{N}^\varepsilon = \left(\mathcal{N} - \bar{\mathcal{N}}\right)$, we have:*

(1) (ε-coverage): $\forall N \in \mathcal{N}^\varepsilon$ $(\varepsilon \in \|N\|)$;

(2) (logical coverage): $\forall i \in [1\,..\,n']$ $(\|\bar{N}_i\| \supseteq \|N'_i\|)$;

(3) (temporal coverage): For each path $\bar{N}_i \rightsquigarrow \bar{N}_j = \langle \bar{N}_i, N^\varepsilon_1, \ldots, N^\varepsilon_s, \bar{N}_j \rangle$ in $\mathcal{O}$, where $\bar{N}_i \in \bar{\mathcal{N}}$, $\bar{N}_j \in \bar{\mathcal{N}}$, $s \geq 0$, $\forall k \in [1\,..\,s]$ $\left(N^\varepsilon_k \in \mathcal{N}^\varepsilon\right)$, the following holds in $\mathcal{O}'$: $N'_i \prec N'_j$.

Example 5. With reference to the observations displayed in Fig. 1, it is easy to show that $\mathcal{O}_1 \trianglerighteq \mathcal{O}_2$. Assume the subset of $\mathcal{N}_1$ being $\bar{\mathcal{N}}_1 = \{N_1, N_2, N_4, N_5\}$. Hence, $\mathcal{N}_1^\varepsilon = \{N_3\}$. Clearly, ε-coverage holds, as $\varepsilon \in \|N_3\|$. Logical coverage holds too, as $\|N_2\| \supseteq \|N'_1\|$, $\|N_1\| \supseteq \|N'_2\|$, $\|N_4\| \supseteq \|N'_3\|$, and $\|N_5\| \supseteq \|N'_4\|$. It is easy to check that temporal coverage occurs. For instance, for $\langle N_1, N_3, N_5 \rangle$, where $N_3 \in \mathcal{N}_1^\varepsilon$, we have $N'_2 \prec N'_4$ in $\mathcal{O}_2$.

Theorem 2 and Note 1 offer evidence that coverage is only sufficient for subsumption, not necessary.

However, in practice, if coverage does not hold, it is unlikely for subsumption to hold. Note that, since coverage entails subsumption, the conditions in Theorem 1 are necessary for coverage too.

Theorem 2. *Coverage entails subsumption:*

$$\mathcal{O} \trianglerighteq \mathcal{O}' \implies \mathcal{O} \Supset \mathcal{O}'. \tag{10}$$

Proof. The proof is based on Definition 1 and Definition 2 (the latter given below).

Definition 2. (Sterile sequence) *Let* $\tilde{\bar{\mathcal{N}}} = \langle \tilde{N}_1, \ldots, \tilde{N}_{n'} \rangle$ *be an ordering*[5] *of nodes in* $\bar{\mathcal{N}}$. *The* sterile sequence *of* $\tilde{\bar{\mathcal{N}}}$,

$$\left\langle \mathcal{N}_0^{\varepsilon}, \mathcal{N}_1^{\varepsilon}, \ldots, \mathcal{N}_{n'}^{\varepsilon} \right\rangle \tag{11}$$

is a sequence of subsets of $\mathcal{N}^{\varepsilon}$, *called* sterile sets, *inductively defined as follows:*

- (Basis) $\mathcal{N}_0^{\varepsilon}$ *is defined by the following two rules:*
  (1) If $N \in \mathcal{N}^{\varepsilon}$, *N is a root of* $\mathcal{O}$, *then* $N \in \mathcal{N}_0^{\varepsilon}$,
  (2) If $N \in \mathcal{N}^{\varepsilon}$, *all parents of N are in* $\mathcal{N}_0^{\varepsilon}$, *then* $N \in \mathcal{N}_0^{\varepsilon}$;
- (Induction) *Given* $\mathcal{N}_i^{\varepsilon}$, $i \in [0 \,..\, (n'-1)]$, *the successive sterile set* $\mathcal{N}_{i+1}^{\varepsilon}$ *is defined by the following two rules:*
  (3) If $N \in \mathcal{N}^{\varepsilon}$, *all parents of N are in* $\left(\mathcal{N}_i^{*} \cup \left\{\tilde{N}_{i+1}\right\}\right)$, *then* $N \in \mathcal{N}_{i+1}^{\varepsilon}$,
  (4) If $N \in \mathcal{N}^{\varepsilon}$, *all parents of N are in* $\left(\mathcal{N}_i^{*} \cup \left\{\tilde{N}_{i+1}\right\} \cup \mathcal{N}_{i+1}^{\varepsilon}\right)$, *then* $N \in \mathcal{N}_{i+1}^{\varepsilon}$,

where $\mathcal{N}_i^{*}$, $i \in [0 \,..\, n']$, *is recursively defined as follows:*

$$\mathcal{N}_i^{*} = \begin{cases} \mathcal{N}_0^{\varepsilon} & \text{if } i = 0 \\ \mathcal{N}_{i-1}^{*} \cup \left\{\tilde{N}_i\right\} \cup \mathcal{N}_i^{\varepsilon} & \text{otherwise.} \end{cases} \tag{12}$$

To prove the theorem, it suffices to show that each candidate signature $\mathbb{S}$ in the index space of $\mathcal{O}'$ is also a candidate signature in the index space of $\mathcal{O}$, namely:

$$\forall \mathbb{S} \in \|Isp(\mathcal{O}')\| \; (\mathbb{S} \in \|Isp(\mathcal{O})\|). \tag{13}$$

According to Theorem 1 in (Lamperti and Zanella, 2002), $\mathbb{S}$ is the sequence of labels obtained by selecting, without violating the precedence constraints of $\mathcal{A}'$, one label from each node in $\mathcal{N}'$, and by removing all the null labels ε. Let

$$\mathbb{N}' = \langle \tilde{N}'_1, \ldots, \tilde{N}'_{n'} \rangle \tag{14}$$

be the ordering of $\mathcal{N}'$ relevant to the choices of such labels. Accordingly, the sequence $\mathbb{L}'$ of the chosen labels can be written as

$$\mathbb{L}' = \langle \ell \mid \ell \in \|\tilde{N}'_i\|, i \in [1 \,..\, n'] \rangle \tag{15}$$

[5] An ordering of a set is a sequence involving all and only the elements in the set, without duplicates.

while the candidate signature $\mathbb{S}$ is in fact

$$\mathbb{S} = \langle \ell \mid \ell \in \mathbb{L}', \ell \neq \varepsilon \rangle. \tag{16}$$

We need to show that there exists an ordering $\mathbb{N}$ of $\mathcal{N}$ fulfilling the precedence constraints imposed by $\mathcal{A}$, from which it is possible to select a sequence $\mathbb{L}$ of labels,

$$\mathbb{L} = \langle \ell_1, \ell_2, \ldots, \ell_n \rangle \tag{17}$$

such that the subsequence of non-null labels in $\mathbb{L}$ equals $\mathbb{S}$:

$$\langle \ell \mid \ell \in \mathbb{L}, \ell \neq \varepsilon \rangle = \mathbb{S}. \tag{18}$$

Note how $\mathbb{N}$ (as well as any other ordering of $\mathcal{N}$) can be represented as a sequence of nodes in $\mathcal{N}$, with each node being interspersed with (possibly empty) subsequences $\mathbb{N}_i^{\varepsilon}$ of nodes in $\mathcal{N}^{\varepsilon}$, specifically

$$\mathbb{N} = \mathbb{N}_0^{\varepsilon} \cup \langle \tilde{N}_1 \rangle \cup \mathbb{N}_1^{\varepsilon} \cup \langle \tilde{N}_2 \rangle \cup \mathbb{N}_2^{\varepsilon} \ldots \langle \tilde{N}_{n'} \rangle \cup \mathbb{N}_{n'}^{\varepsilon} \tag{19}$$

where

$$\bigcup_{i=1}^{n'} \{\tilde{N}_i\} = \bar{\mathcal{N}}, \quad \bigcup_{i=0}^{n'} \mathbb{N}_i^{\varepsilon} = \mathcal{N}^{\varepsilon}, \quad \bigcap_{i=0}^{n'} \mathbb{N}_i^{\varepsilon} = \emptyset. \tag{20}$$

The proof is by induction on $\mathbb{L}'$. Let $\mathbb{L}'_i$ denote the subsequence of $\mathbb{L}'$ up to the i-th label, $i \in [1 \,..\, n']$. Let $\mathbb{L}_i$ denote the subsequence of $\mathbb{L}$ relevant to the choices of labels performed in correspondence of the labels in $\mathbb{L}'_i$. Let $\mathbb{S}_i$ and $\mathbb{S}'_i$ denote the candidate signatures corresponding to $\mathbb{L}_i$ and $\mathbb{L}'_i$, respectively.[6]

(*Basis*) No label is chosen in $\mathcal{O}'$, that is, $\mathbb{L}'_0 = \langle\rangle$. We choose a sequence of empty labels for all the nodes in $\mathcal{N}_0^{\varepsilon}$, which is clearly possible according to the property that $\mathcal{N}_0^{\varepsilon}$ is a sterile set composed of nodes having ancestors in $\mathcal{N}^{\varepsilon}$ only. In other words, $\mathbb{N}_0^{\varepsilon}$ is an ordering of $\mathcal{N}_0^{\varepsilon}$, while $\mathbb{L}_0 = \langle \varepsilon, \varepsilon, \ldots, \varepsilon \rangle$, hence, $\mathbb{S}_0 = \mathbb{S}'_0 = \langle\rangle$.

(*Induction*) We assume that $\mathbb{L}_i$ and $\mathbb{L}'_i$, $i \in [0 \,..\, (n'-1)]$, are such that $\mathbb{S}_i = \mathbb{S}'_i$. We also assume that, given the sequence $\langle \tilde{N}'_1, \ldots, \tilde{N}'_i \rangle$ of chosen nodes in $\mathcal{O}'$, the corresponding sequence of chosen nodes in $\mathcal{O}$ is $\mathbb{N}_0^{\varepsilon} \cup \langle \tilde{N}_1 \rangle \cup \mathbb{N}_1^{\varepsilon} \cup \langle \tilde{N}_2 \rangle \cup \mathbb{N}_2^{\varepsilon} \ldots \langle \tilde{N}_i \rangle \cup \mathbb{N}_i^{\varepsilon}$, where, $\forall k \in [1 \,..\, i]$, if $\tilde{N}'_k$ is the node N'_h in $\mathcal{N}'$, then $\tilde{N}_k$ is the node $\bar{N}_h$ in $\bar{\mathcal{N}}$, and each $\mathbb{N}_k^{\varepsilon}$ is an ordering of $\mathcal{N}_k^{\varepsilon}$. We have to show that, once chosen the next label $\ell \in \|\tilde{N}'_{i+1}\|$, thereby determining $\mathbb{L}'_{i+1}$ and $\mathbb{S}'_{i+1}$, it is possible to choose a node $\tilde{N}_{i+1} \in \bar{\mathcal{N}}$ that includes ℓ, and $\mathbb{N}_{i+1}^{\varepsilon}$ as an ordering of $\mathcal{N}_{i+1}^{\varepsilon}$ from which ε is chosen, thereby determining $\mathbb{L}_{i+1}$ such that $\mathbb{S}_{i+1} = \mathbb{S}'_{i+1}$.

[6] Note how $\mathbb{L}'_i$ includes exactly i labels, while, owing to the ε selected for nodes in $\mathcal{N}_{\varepsilon}$, the number of labels in $\mathbb{L}_i$ is possibly greater than i.

Let N'_j be the node in $\mathcal{N}' = \{N'_1, \ldots, N'_{n'}\}$ corresponding to $\tilde{N}'_{i+1}$. According to logical coverage in Definition 1, there exists a node $\bar{N}_j$ in $\bar{\mathcal{N}} = \{\bar{N}_1, \ldots, \bar{N}_{n'}\}$ such that $\|\bar{N}_j\| \supseteq \|N'_j\|$, in other words, $\bar{N}_j$ includes ℓ. We consider $\tilde{N}_{i+1} = \bar{N}_j$. In order for $\bar{N}_j$ to be actually chosen, we have to show that each parent node N of $\bar{N}_j$ in $\mathcal{O}$ was already considered, that is, N belongs to the prefix of $\tilde{\mathcal{N}}$ relevant to $\mathbb{L}_i$. Two cases are possible for N:

(a) N is a node $\bar{N}_k \in \bar{\mathcal{N}}$. On the one hand, owing to temporal coverage, $\bar{N}_k \mapsto \bar{N}_j$ in $\mathcal{O}$ entails $N'_k \prec N'_j$ in $\mathcal{O}'$. On the other, since N'_j was chosen in $\mathcal{O}'$, all its parent nodes must have been considered already, that is, $N'_k \in \langle \tilde{N}'_1, \ldots, \tilde{N}'_i \rangle$. Since, based on the assumption of *Induction*, we always choose for each node in $N'_m \in \mathcal{N}'$ the corresponding node $\bar{N}_m \in \bar{\mathcal{N}}$, it is possible to claim that $\bar{N}_k$ was already considered in $\mathcal{O}$, that is, $\bar{N}_k \in \langle \tilde{N}_1, \ldots, \tilde{N}_i \rangle$.

(b) $N \in \mathcal{N}_\varepsilon$. We consider each path $N_{\mathrm{a}} \rightsquigarrow N$ in $\mathcal{O}$ such that N_{a} is the first ancestor of N (possibly N itself), where either N_{a} is a root of $\mathcal{O}$ or $N_{\mathrm{a}} \in \bar{\mathcal{N}}$. Let $\mathcal{N}_{\mathrm{a}}$ be the set of such ancestors. We show that each node $N_{\mathrm{a}} \in \mathcal{N}_{\mathrm{a}}$ has been considered already. Two cases are possible: either $N_{\mathrm{a}} \in \mathcal{N}_\varepsilon$ or $N_{\mathrm{a}} \in \bar{\mathcal{N}}$. In the first case, N_{a} is a node in the sterile set $\mathcal{N}_0^\varepsilon$ and, hence, it has been considered in $\mathbb{N}_0^\varepsilon$ already (see *Basis*). In the second case ($N_{\mathrm{a}} \in \bar{\mathcal{N}}$), let $\bar{N}_h$ be the node in $\bar{\mathcal{N}}$ corresponding to N_{a}. We consider a path $\bar{N}_h \rightsquigarrow N \mapsto \bar{N}_j$. Since between $\bar{N}_h$ and $\bar{N}_j$ are only nodes in $\mathcal{N}_\varepsilon$, temporal coverage implies that $N'_h \prec N'_j$ in $\mathcal{O}'$, where N'_h is the node in $\mathcal{N}'$ corresponding to $\bar{N}_h$. Thus, N'_h was already considered in $\mathcal{O}'$. As, based on the assumption in *Induction*, we always choose in $\mathcal{O}$ the corresponding node of that chosen in $\mathcal{O}'$, this implies that $\bar{N}_h$ was already considered in $\mathcal{O}$ too. We conclude that all nodes in $\mathcal{N}_{\mathrm{a}}$ have been considered. Now, it is clear that N is either in $\mathcal{N}_{\mathrm{a}}$ or N is a node belonging to the sterile set of some node in $\mathcal{N}_{\mathrm{a}}$. In either case, owing to the assumption of *Induction*, N must have been considered already. In other words, all parents of $\bar{N}_j$ have been chosen already, thereby allowing $\bar{N}_j$ itself, alias $\tilde{N}_{i+1}$, to be chosen. Furthermore, based on the definition of sterile sequence, we may also consider an ordering $\mathbb{N}_{i+1}^\varepsilon$ of $\mathcal{N}_{i+1}^\varepsilon$ and choose label ε for each of such nodes, thereby leading to the conclusion that $\mathbb{S}_{i+1} = \mathbb{S}'_{i+1}$. □

Note 1. *Coverage is stronger than subsumption, namely:*

$$\mathcal{O} \Supset \mathcal{O}' \quad \not\Rightarrow \quad \mathcal{O} \trianglerighteq \mathcal{O}'. \tag{21}$$

To be convinced, it suffices to show an example in which subsumption holds while coverage does not. Consider two observations, $\mathcal{O} = (\mathcal{N}, \mathcal{L}, \mathcal{A})$ and $\mathcal{O}' = (\mathcal{N}', \mathcal{L}', \mathcal{A}')$, where $\mathcal{N} = \{N_1, N_2\}$, $\mathcal{N}' = \{N'_1, N'_2\}$, $\mathcal{L} = \mathcal{L}' = \{a\}$, $\mathcal{A} = \{N_1 \mapsto N_2\}$, $\mathcal{A}' = \emptyset$, and $\|N_1\| = \|N_2\| = \|N'_1\| = \|N'_2\| = \{a\}$, as displayed in Fig. 3.

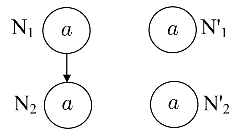

Figure 3: Observations $\mathcal{O}$ (left) and $\mathcal{O}'$ (right).

Clearly, $\bar{\mathcal{N}} = \mathcal{N}$ and $\mathcal{N}_\varepsilon = \emptyset$. Note how, unlike $\mathcal{O}$, since $\mathcal{A}' = \emptyset$, $\mathcal{O}'$ does not force any temporal constraint between N_1 and N_2. Incidentally, both observations involve just one candidate signature, namely $\mathbb{S} = \langle a, a \rangle$. Thus, since $\|Isp(\mathcal{O})\| = \|Isp(\mathcal{O}')\| = \{\langle a, a \rangle\}$, both observations subsume each other, in particular $\mathcal{O} \Supset \mathcal{O}'$. However, it is easy to realize that O does *not* cover $\mathcal{O}'$, namely $\mathcal{O} \not\trianglerighteq \mathcal{O}'$. In fact, due to the symmetry of $\mathcal{O}'$, we can choose any of the two possible associations between nodes in $\mathcal{O}$ and nodes in $\mathcal{O}'$, for instance, $\bar{\mathcal{N}} = \{N_1, N_2\}$. Based on Definition 1, on the one hand, both ε-coverage and logical coverage hold. On the other, temporal coverage is missing, as for $N_1 \mapsto N_2$ in $\mathcal{O}$, we have $N'_1 \not\prec N'_2$. The same negative result occurs for $\bar{\mathcal{N}} = \{N_2, N_1\}$. In other terms, $\mathcal{O} \not\trianglerighteq \mathcal{O}'$.

4 TESTING COVERAGE

In this section we give an abstract, pseudo-coded implementation of subsumption-checking via coverage. Specifically, Algorithm 1 tests both the necessary conditions of Theorem 1 and the coverage relationship. A tracing of the algorithm on observations in Fig. 1 is provided in Example 6.

Algorithm 1. (COVERS) The *Covers* function (lines 1–41) takes as input two observations, $\mathcal{O}$ and $\mathcal{O}'$, and outputs a Boolean value indicating whether or not $\mathcal{O}$ covers $\mathcal{O}'$. The body of *Covers* is outlined in lines 30–41. In lines 31–32, the observation parameters for $\mathcal{O}$ and $\mathcal{O}'$ are set. Then, at line 33, conditions (5) and (6) of Theorem 1, along with condition (9) of Corollary 1.1, are checked. In lines 36–38, the multisets $\mathcal{M}$ and $\mathcal{M}'$ of instances of labels are created, with the former decremented by $d = (n - n')$ instances of label ε, which is the cardinality of $(\mathcal{N} - \mathcal{N}')$. This allows the algorithm to retain a sufficient number of spare nodes in $\mathcal{N}$ that contain ε, namely $\mathcal{N}^\varepsilon$ in Definition 1. At line 39, condition (7) of Theorem 1 is checked. The algorithm yields $\overline{\mathcal{N}}$, the subset of $\mathcal{N}$ that is associated with $\mathcal{N}'$ in Definition 1, by building the set $\mathcal{R}$ of associations through the call to the auxiliary function *CovStep* at line 40. The specification of *CovStep* is given in lines 3–29. Besides $\mathcal{O}$, $\mathcal{O}'$, $\mathcal{M}$, and $\mathcal{M}'$, it takes as input $\mathcal{C}$ and $\mathcal{C}'$, the set of nodes already considered in $\mathcal{O}$ and $\mathcal{O}'$, respectively, along with d, the number of nodes in $\mathcal{N}^{\varepsilon}$ not yet considered, and $\mathcal{R}$, the set of associations made up so far. The body of *CovStep* starts at line 10, where the cardinality of $\mathcal{R}$ is tested: if $\mathcal{R}$ contains n' pairs, it means that all nodes in $\mathcal{N}'$ have been considered and $\mathcal{N}$ is completed, thereby, coverage holds. Otherwise, a new node N' in $\mathcal{O}'$ is considered at line 11, such that all its parent nodes have been considered already. At line 12, the set $\mathcal{F}$ of nodes in $\mathcal{O}$ is created, which includes the unconsidered nodes of $\mathcal{O}$ with all parents already in $\mathcal{C}$. A loop for each node N in $\mathcal{F}$ is iterated in lines 13–27. First, logical coverage and containment relationship of labels are tested (line 14). Then, the set $\mathcal{N}_a$ of the nearest ancestors[7] of N which have been already involved in the associations of $\mathcal{R}$ is instantiated (line 15). This allows temporal-coverage checking (line 16). If the latter succeeds, *CovStep* is recursively called at line 17, with new actual parameters: the sets $\mathcal{C}$ and $\mathcal{C}'$ of considered nodes are extended with N and N', respectively, the multisets $\mathcal{M}$ and $\mathcal{M}'$ are decremented by the labels in N and N', respectively, while $\mathcal{R}$ is extended with the new pair (N, N'). If such a call succeeds, the current activation of *CovStep* succeeds too (line 18). If not, or either logical or temporal coverage fails, a chance still remains by assuming $N \in \mathcal{N}^{\varepsilon}$: this is viable only on condition that N include ε, there exists at least one spare node in $\mathcal{N}^{\varepsilon}$ ($d > 0$), and the multiset $\mathcal{M}$ contains $\mathcal{M}'$ once decremented by the labels of N, ε aside (line 22)[8]. If so, a different recursive call to *CovStep* is performed at line 23, with the changed parameters being the (extended) set $\mathcal{C}$ of consumed nodes in $\mathcal{O}$, the (decremented) multiset $\mathcal{M}$, and the decremented value of d. If such a call succeeds, the current activation of *CovStep* succeeds too. If not, the loop is iterated and a new node in $\mathcal{F}$ is tried. If the computation exits the loop in a natural way, it means that no node can be associated with N' within this computational context, thereby causing the current activation of *CovStep* to fail (line 28).

```
1.  function $Covers(\mathcal{O},\mathcal{O}')$: Bool
2.    $\mathcal{O}=(\mathcal{N},\mathcal{L},\mathcal{A})$, $\mathcal{O}'=(\mathcal{N}',\mathcal{L}',\mathcal{A}')$: observations;
3.    function $CovStep(\mathcal{O},\mathcal{O}',\mathcal{C},\mathcal{C}',\mathcal{M},\mathcal{M}',d,\mathcal{R})$: Bool
4.      $\mathcal{O}=(\mathcal{N},\mathcal{L},\mathcal{A})$, $\mathcal{O}'=(\mathcal{N}',\mathcal{L}',\mathcal{A}')$: observations,
5.      $\mathcal{C},\mathcal{C}'$: the set of consumed nodes for $\mathcal{O}$ and $\mathcal{O}'$,
6.      $\mathcal{M},\mathcal{M}'$: the multisets of labels in $\mathcal{O}$ and $\mathcal{O}'$,
7.      $d$: the number of nodes in $\mathcal{N}$ that can still be in $\mathcal{N}^{\varepsilon}$,
8.      $\mathcal{R}\subseteq\mathcal{N}\times\mathcal{N}'$: a relation on $\mathcal{N}$ and $\mathcal{N}'$;
9.    begin {CovStep}
10.     if $|\mathcal{R}|=n'$ then return true end-if;
11.     Pick up a node $N'\in(\mathcal{N}'-\mathcal{C}')$ with parents in $\mathcal{C}'$;
12.     $\mathcal{F}:=\{N \mid N\in(\mathcal{N}-\mathcal{C})$, all parents of $N$ are in $\mathcal{C}\}$;
13.     for each $N\in\mathcal{F}$ do
14.       if $\|N\|\supseteq\|N'\| \wedge (\mathcal{M}-\|N\|)\supseteq(\mathcal{M}'-\|N'\|)$ then
15.         $\mathcal{N}_a$ := the set of nearest ancestors of $N$ in $\mathcal{R}(\mathcal{N})$;
16.         if $\forall N_a\in\mathcal{N}_a,(N_a,N'_a)\in\mathcal{R}\ (N'_a\prec N')$ then
17.           if $CovStep(\mathcal{O},\mathcal{O}',\mathcal{C}\cup\{N\},\mathcal{C}'\cup\{N'\},\mathcal{M}-\|N\|,$
                  $\mathcal{M}'-\|N'\|,d,\mathcal{R}\cup\{(N,N')\})$ then
18.             return true
19.           end-if
20.         end-if
21.       end-if;
22.       if $\varepsilon\in\|N\| \wedge d>0 \wedge (\mathcal{M}-(\|N\|-\{\varepsilon\}))\supseteq\mathcal{M}'$ then
23.         if $CovStep(\mathcal{O},\mathcal{O}',\mathcal{C}\cup\{N\},\mathcal{C}',$
                  $\mathcal{M}-(\|N\|-\{\varepsilon\}),\mathcal{M}',d-1,\mathcal{R})$ then
24.           return true
25.         end-if
26.       end-if
27.     end-for;
28.     return false
29.   end {CovStep};
30. begin{Covers}
31.   $n:=|\mathcal{N}|$; $n_{\varepsilon}:=\{N \mid N\in\mathcal{N},\varepsilon\in\|N\|\}$;
32.   $n':=|\mathcal{N}'|$; $n'_{\varepsilon}:=\{N' \mid N'\in\mathcal{N}',\varepsilon\in\|N'\|\}$;
33.   if $n<n' \vee n_{\varepsilon}-n'_{\varepsilon}<n-n' \vee \mathcal{L}\not\supseteq\mathcal{L}'$ then
34.     return false
35.   end-if:
36.   Create the multisets $\mathcal{M}$ and $\mathcal{M}'$ of labels in $\mathcal{O}$, $\mathcal{O}'$;
37.   $d:=n-n'$;
38.   Remove $d$ instances of label $\varepsilon$ from $\mathcal{M}$;
39.   if $\mathcal{M}\not\supseteq\mathcal{M}'$ then return false end-if;
40.   return $CovStep(\mathcal{O},\mathcal{O}',\emptyset,\emptyset,\mathcal{M},\mathcal{M}',d,\emptyset)$
41. end {Covers}.
```

Example 6. With reference to the observations in Fig. 1, consider the run of $Covers(\mathcal{O}_1, \mathcal{O}_2)$. Since, according to Example 4, all the necessary conditions of Theorem 1 hold, we focus our attention on the first call to *CovStep* at line 40.

Depicted in Fig. 4 is the tree of the recursive activations to *CovStep*, where each node i corresponds to the i-th call (dashed nodes correspond to calls at line 23, with the others corresponding to line 17).

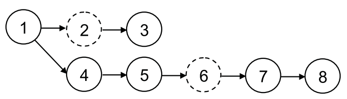

Figure 4: Activation tree for *CovStep* in Example 6.

[7]The nearest ancestors of a node are not necessarily its parents, since a parent node may not belong to $\mathcal{R}(\mathcal{N})$, as it is included in N^{ε}.

[8]When a spare node is consumed, ε is retained in $\mathcal{M}$ because. at line 38, all instances of ε relevant to spare nodes were removed from $\mathcal{M}$ already.

Table 1: Tracing of $Covers(\mathcal{O}_1, \mathcal{O}_2)$ in Example 6.

Id	$\mathcal{C}$	$\mathcal{C}'$	$\mathcal{M}$	$\mathcal{M}'$	d	$\mathcal{R}$
1	$\emptyset$	$\emptyset$	$\{a,a,a,b,b,b,b,c,d,f,\varepsilon,\varepsilon\}$	$\{a,a,b,c,d,\varepsilon,\varepsilon\}$	1	$\emptyset$
2	$\{1\}$	$\emptyset$	$\{a,a,b,b,b,c,d,f,\varepsilon,\varepsilon\}$	$\{a,a,b,c,d,\varepsilon,\varepsilon\}$	0	$\emptyset$
3	$\{1,2\}$	$\{1'\}$	$\{a,b,b,c,d,f,\varepsilon,\varepsilon\}$	$\{a,b,c,d,\varepsilon,\varepsilon\}$	0	$\{(2,1')\}$
4	$\{2\}$	$\{1'\}$	$\{a,a,b,b,b,c,d,f,\varepsilon,\varepsilon\}$	$\{a,b,c,d,\varepsilon,\varepsilon\}$	1	$\{(2,1')\}$
5	$\{1,2\}$	$\{1',2'\}$	$\{a,b,b,c,d,f,\varepsilon\}$	$\{a,c,d,\varepsilon\}$	1	$\{(2,1'),(1,2')\}$
6	$\{1,2,3\}$	$\{1',2'\}$	$\{a,b,b,c,d,\varepsilon\}$	$\{a,c,d,\varepsilon\}$	0	$\{(2,1'),(1,2')\}$
7	$\{1,2,3,4\}$	$\{1',2',3'\}$	$\{a,b,\varepsilon\}$	$\{a,\varepsilon\}$	0	$\{(2,1'),(1,2'),(4,3')\}$
8	$\{1,2,3,4,5\}$	$\{1',2',3',4'\}$	$\emptyset$	$\emptyset$	0	$\{(2,1'),(1,2'),(4,3'),(5,4')\}$

Relevant details are given in Table 1, with *Id* being the identifier of the call, while the other columns indicate the actual parameters of the call (observation nodes are identified by the corresponding subscripts). The computation is described by the following steps, where item numbers stand for activation identifiers, namely *Id*.

1. $N' = 1'$, $\mathcal{F} = \{1,2\}$. Within the loop (line 13), choosing $N = 1$ makes the multiset containment false (line 14). However, since condition at line 22 holds for N, a recursive call to *CovStep* is performed at line 23 (see *Id* $= 2$ in Table 1).
2. $N' = 1'$, $\mathcal{F} = \{2,3\}$. With $N = 2$, a recursive call is performed at line 17 (*Id* $= 3$ in Table 1).
3. $N' = 2'$, $\mathcal{F} = \{3,4\}$. With $N = 3$, logical coverage fails, as $\|N\| \not\supseteq \|N'\|$. Besides, although $\varepsilon \in \|N\|$, condition at line 22 is false because $d = 0$ (no further spare nodes to assume in $\mathcal{N}^{\varepsilon}$). Thus, a new iteration of loop at line 13 is performed with $N = 4$: logical coverage fails, while condition at line 22 is false (since $d = 0$ and $\varepsilon \notin \|N\|$). This causes the control to return to the second call, where condition at line 22 is false. Therefore, a new iteration of loop at line 13 is performed, now with $N = 3$. Since both checks at lines 14 and 22 fail, the control returns to the first call, where $N = 2$ is chosen: this allows the fourth recursive call at line 17 (*Id* $= 4$).
4. $N' = 2'$, $\mathcal{F} = \{1,4\}$. With $N = 1$, a recursive call is performed at line 17 (*Id* $= 5$).
5. $N' = 3'$, $\mathcal{F} = \{3,4\}$. With $N = 3$, logical coverage fails. However, since condition at line 22 holds, a recursive call is performed at line 23 (*Id* $= 6$).
6. $N' = 3'$, $\mathcal{F} = \{4\}$. With $N = 4$, a recursive call is performed at line 17 (*Id* $= 7$).
7. $N' = 4'$, $\mathcal{F} = \{5\}$. With $N = 5$, a recursive call is performed at line 17 (*Id* $= 8$).
8. At line 10, since $|\mathcal{R}| = 4$, *CovStep* succeeds.

Proposition 1. *Algorithm 1 is a sound and complete implementation of coverage:*

$$Covers(\mathcal{O}, \mathcal{O}') \iff \mathcal{O} \trianglerighteq \mathcal{O}'. \qquad (22)$$

Proof (sketch). To prove equivalence (22), we first show

$$Covers(\mathcal{O}, \mathcal{O}') \implies \mathcal{O} \trianglerighteq \mathcal{O}'. \qquad (23)$$

Assuming $Covers(\mathcal{O}, \mathcal{O}')$ succeeding means that the call to *CovStep* at line 40 returns **true**. Function *CovStep* recursively instantiates the set $\mathcal{R}$ of associations of nodes (N, N'), for which both logical coverage (line 14) and temporal coverage (line 16), required by Definition 1, hold. Moreover, ε-coverage is supported by conditions at line 22 and the initialization at lines 36–38, which allow for retaining the $(n - n')$ nodes of $\mathcal{N}^{\varepsilon}$ once $\mathcal{R}$ is completed (line 10). In other words, entailment (23) holds. Then, we have to show

$$\mathcal{O} \trianglerighteq \mathcal{O}' \implies Covers(\mathcal{O}, \mathcal{O}'). \qquad (24)$$

The proof is by contradiction. Assume that $\mathcal{O} \trianglerighteq \mathcal{O}'$, while $Covers(\mathcal{O}, \mathcal{O}') =$ **false**. Based on Definition 1, let $\mathcal{R}^* = \{(\bar{N}_1, N'_1), \ldots, (\bar{N}_{n'}, N'_{n'})\}$ denote the relation between $\mathcal{N}$ and $\mathcal{N}'$. Based on a run of *Covers*, we show that *Covers* necessarily makes up $\mathcal{R} = \mathcal{R}^*$. The proof is by induction on $\mathcal{R}$. Note how we can restrict our analysis to the recursive call to *CovStep*, as lines 31–39 simply check the necessary conditions of subsumption stated by Theorem 1 and Corollary 1.1. In fact, since coverage entails subsumption (Theorem 2), such conditions are necessary for coverage too, in other words, the computation surely reaches the call to *CovStep* at line 40. Moreover, such call is supposed to return **false** (owing to the assumption $Covers(\mathcal{O}, \mathcal{O}') =$ **false**).

(*Basis*) Focus on the first call to *CovStep*, where $\mathcal{C}$, $\mathcal{C}'$, and $\mathcal{R}$ are empty, and consider the (first) node N' chosen at line 11. Let N' correspond to the j-th node in $\mathcal{N}'$, namely N'_j. Let $\bar{N}_j$ be the node in $\mathcal{N}$ associated with N'_j in $\mathcal{R}^*$, namely $(\bar{N}_j, N'_j) \in \mathcal{R}^*$. Based on Definition 1, temporal coverage requires that, for each path $\bar{N}_i \rightsquigarrow \bar{N}_j$ in $\mathcal{O}$, where all intermediate nodes in the path are in $\mathcal{N}^{\varepsilon}$, we have $N'_i \prec N'_j$ in $\mathcal{O}'$. However, none of such paths exists, as $N' = N'_j$ is chosen with $\mathcal{C}' = \emptyset$, that is, N' has no parent nodes. Consequently, all ancestors of $\bar{N}_j$ in $\mathcal{O}$ (if any) are in $\mathcal{N}^{\varepsilon}$, that is, they contain label ε. Since *CovStep* is supposed to fail, it will try all choices of N in $\mathcal{F}$. Two cases are possible: either $\bar{N}_j$ is a root of $\mathcal{O}$ or all ancestors of $\bar{N}_j$ are in $\mathcal{N}^{\varepsilon}$. In the first case, $N = \bar{N}_j$ is associated in $\mathcal{R}$ with $N' = N'_j$ within the recursive call to *CovStep* at line 17. In the second case, the same association will be created after a number of recursive calls of *CovStep* at line 23, as all calls to *CovStep* are assumed to fail (including the one creating such association). Thus, in any case, the first choice of N' will led to an association (N, N') which is in $\mathcal{R}^*$ too.

(*Induction*) Assume, in the current call to *CovStep*, $\mathcal{R} = \{(\bar{N}_{c_1}, N'_{c_1}), \ldots, (\bar{N}_{c_k}, N'_{c_k})\}$, where $\mathcal{R} \subset \mathcal{R}^*$, that is, all associations yielded by *CovStep* are in $\mathcal{R}^*$ too. Let $\mathcal{N}_c$ and $\mathcal{N}_c'$ denote the projections of $\mathcal{R}$ on $\mathcal{N}$ and $\mathcal{N}'$, respectively. Now, consider the next choice of N' at line 11. Let N' correspond to the j-th node in $\mathcal{N}'$, namely N'_j. Let $\bar{N}_j$ be the node in $\mathcal{N}$ associated with N'_j in $\mathcal{R}^*$, namely $(\bar{N}_j, N'_j) \in \mathcal{R}^*$. Based on Definition 1, temporal coverage requires that, for each path $\bar{N}_i \rightsquigarrow \bar{N}_j$ in $\mathcal{O}$, where all intermediate nodes in the path are in $\mathcal{N}^{\varepsilon}$, $N'_i \prec N'_j$ holds in $\mathcal{O}'$. This implies that all N'_i are in $\mathcal{N}_c'$ and, in consequence of the inductive assumption, all $\bar{N}_i$ are in $\mathcal{N}_c$. Hence, following the same argumentation outlined in *Basis*, $\bar{N}_j$ can be considered and associated with N'_j. Thus, $(\bar{N}_j, N'_j)$ is inserted into $\mathcal{R}$. This leads to the claim that $(\mathcal{R} \cup \{(\bar{N}_j, N'_j)\}) \subseteq \mathcal{R}^*$, which concludes the proof of *Induction*. Thus, equation (24) is proved. $\square$

5 EXPERIMENTAL RESULTS

A number of experiments were carried out in order to assess the coverage approach to subsumption checking based on different classes of observations. We ran subsumption checking using two different algorithms prototyped in Haskell functional language (Thompson, 1999), namely *Subsumes* and *Covers*. The former is strictly based on the definition of subsumption and requires testing index-space (automaton) containment. We considered three classes of observations, namely *disconnected*, *connected*, and *linear*. In disconnected observations, no temporal constraints are given among nodes, thereby maximizing temporal uncertainty. Instead, in connected observations, each node is temporally linked with other nodes. Linear observations are a subclass of connected observations where no temporal uncertainty occurs. The experimental results in this paper refer to connected observations. In order to stress the computation, we chose observations for which subsumption hold, so that the necessary conditions in Theorem 1 always hold.[9]

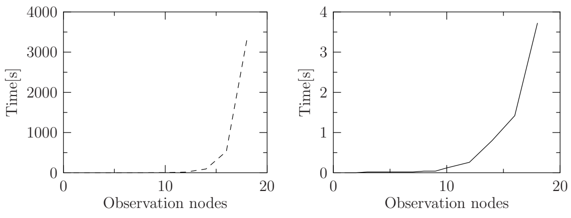

Figure 5: Checking subsumption: time response.

Shown in Fig. 5 is the response time for the two algorithms, with the x-axis marked by the number of nodes in the involved observations. Precisely, the y-axis indicates the time for *Subsumes* (dashed line, on the left) and *Covers* (plain line, on the right) to emit the relevant verdict. Considering the different scale of the y-axis, the comparison is striking in favor of *Covers*. Displayed in Fig. 6 is the maximum space allocation for the two algorithms, which shows how no considerable difference exists between them.

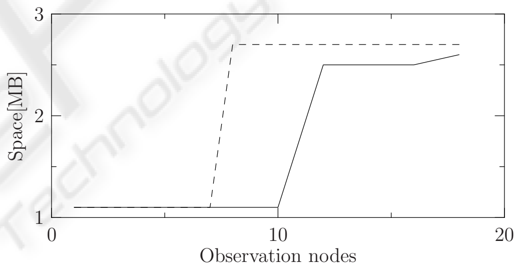

Figure 6: Checking subsumption: space allocation.

6 CONCLUSIONS

A technique for checking observation-subsumption in diagnosis of DESs has been proposed. This check is required to pursue similarity-based diagnosis, where the solution of a diagnostic problem is possibly supported by the solution of a previously-solved problem stored in a knowledge-base. The solution to such checking-problem can be provided strictly based on the definition of observation-subsumption, which requires the generation and comparison of the index spaces of the two observations, where an index space is an acyclic automaton. Since index-space generation and processing are computationally complex, an alternative technique has been envisaged and formally defined in this paper, which exploits a number of necessary conditions, as well as a sufficient condition, for

[9]In fact, when one of such conditions is violated, *Covers* is increasingly more efficient than *Subsumes*.

subsumption to hold. The latter is based on the notion of coverage, which allows the direct comparison of the two observations without any index-space generation or manipulation. The new approach has been tested and compared with the previous (systematic) approach. Experimental results indicate that the technique is considerably worthwhile as to time complexity. However, since the implementation is based on a pure functional language, chances are that implementing it through a more efficient general-purpose language is bound to still better figures.

REFERENCES

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

Brand, D. and Zafiropulo, P. (1983). On communicating finite-state machines. *Journal of ACM*, 30(2):323–342.

Cassandras, C. and Lafortune, S. (1999). *Introduction to Discrete Event Systems*, volume 11 of *The Kluwer International Series in Discrete Event Dynamic Systems*. Kluwer Academic Publisher, Boston, MA.

Cerutti, S., Lamperti, G., Scaroni, M., Zanella, M., and Zanni, D. (2007). A diagnostic environment for automaton networks. *Software – Practice and Experience*, 37(4):365–415. DOI: 10.1002/spe.773.

Chen, Y. and Provan, G. (1997). Modeling and diagnosis of timed discrete event systems - a factory automation example. In *American Control Conference*, pages 31–36, Albuquerque, NM.

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

Debouk, R., Lafortune, S., and Teneketzis, D. (2000). Coordinated decentralized protocols for failure diagnosis of discrete-event systems. *Journal of Discrete Event Dynamic Systems: Theory and Application*, 10:33–86.

Hopcroft, J., Motwani, R., and Ullman, J. (2006). *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, Reading, MA, third edition.

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. (2004). 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. (2006a). Flexible diagnosis of discrete-event systems by similarity-based reasoning techniques. *Artificial Intelligence*, 170(3):232–297.

Lamperti, G. and Zanella, M. (2006b). Incremental processing of temporal observations in supervision and diagnosis of discrete-event systems. In *Eighth International Conference on Enterprise Information Systems – ICEIS'2006*, pages 47–57, Paphos, Cyprus.

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. (2000). Decentralized diagnoser approach: application to telecommunication networks. In *Eleventh International Workshop on Principles of Diagnosis – DX'00*, pages 185–192, Morelia, MX.

Pencolé, Y. and Cordier, M. (2005). A formal framework for the decentralized diagnosis of large scale discrete event systems and its application to telecommunication networks. *Artificial Intelligence*, 164:121–170.

Rozé, L. and Cordier, M. (2002). Diagnosing discrete-event systems: extending the 'diagnoser approach' to deal with telecommunication networks. *Journal of Discrete Event Dynamic Systems: Theory and Application*, 12:43–81.

Sampath, M., Lafortune, S., and Teneketzis, D. (1998). Active diagnosis of discrete-event systems. *IEEE Transactions on Automatic Control*, 43(7):908–929.

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.

Schullerus, G. and Krebs, V. (2001). Diagnosis of a class of discrete-event systems based on parameter estimation of a modular algebraic model. In *Twelfth International Workshop on Principles of Diagnosis – DX'01*, pages 189–196, San Sicario, I.

Thompson, S. (1999). *Haskell – The Craft of Functional Programming*. Addison-Wesley, Harlow, UK.

Zad, S., Kwong, R., and Wonham, W. (1999). Fault diagnosis in timed discrete-event systems. In *38th IEEE Conference on Decision and Control – CDC'99*, pages 1756–1761, Pheonix, AZ. IEEE, Piscataway, NJ.