Proceedings of the Sixteenth International Conference on
Principles of Knowledge Representation and Reasoning (KR 2018)

Abductive Diagnosis of Complex Active Systems with Compiled Knowledge

Gianfranco Lamperti, Marina Zanella
Department of Information Engineering,
University of Brescia,
Via Branze 38, 25123, Brescia, Italy

Xiangfu Zhao
College of Mathematics, Physics, and Information
Engineering, Zhejiang Normal University,
688 Yingbin Road, Jinhua City,
Zhejiang Province, 321004, P.R. China

Abstract

An abduction-based diagnosis technique for a class of discrete-event systems (DESs), called complex active systems (CASs), is presented. Inspired by complexity science, according to which the essence of a complex system is the emergence of unpredictable behavior from interaction among components, the behavior of a CAS is stratified in a hierarchy of active units (AUs), where an AU is a sort of DES involving a set of interconnected components that are modeled as communicating automata. The collective interaction among components within an AU gives rise to the occurrence of emergent events that may affect the behavior of a superior AU. This allows for the modeling of a system at different abstraction levels, where emergent events are the means of communication between different layers of the hierarchy. In order to speed up the online diagnosis task, model-based reasoning on deep knowledge of the CAS is performed offline. This results in a more abstract compiled knowledge, which is then exploited online by the diagnosis engine based on a given temporal observation of the CAS. The correctness (soundness and completeness) of the technique is formally proven.

1 Introduction

Colloquially, the qualifier "complex" indicates something that is complicated in nature. In this perspective, a *complex system* is complicated in structure and behavior, such as an aircraft or a nuclear power plant. However, according to complexity science (Atay and Jost 2004; Bossomaier and Green 2007; Licata and Sakaji 2008; Goles and Martinez 2001), although it is likely to be complicated, a complex system does not equate to a complicated system. In this different perspective, a complex system consists of several components that, once aggregated, assume collective characteristics that are not manifested, and cannot be predicted from the properties of the individual components. So, a human being is much more than the union of some 100 trillion cells that make up his / her body. Similarly, an ecosystem is much more than the union of its components, namely plants and animals. Consider a colony of ants, which is capable of building a strikingly sophisticated "city" from scratch in a few days, and without the help of a blueprint. Each ant performs very simple actions, such as moving a grain of soil from one place to another. Yet, despite the poor craft-profile of each single ant, the collective result is stunningly impressive, seemingly being designed by a team of architects and engineers. The point is, even if we understand the behavior of each individual ant in terms of its interaction with other ants, nonetheless it is impossible to predict the global behavior of the ant colony, much less its final result. What is missing is the understanding of the colony collective behavior *emerging* from the interaction of the single ants. Complexity science questions the traditional reductionist approach adopted in the natural sciences, namely the reduction of complex natural phenomena to several simple processes, and the application of a simple theory to each process. More specifically, the *principle of superimposition* is no longer accepted, namely that the comprehension of the whole phenomenon relies on the superimposition of its parts. Based on this principle, for instance, if you understand the theory of an elementary particle, then you will understand every natural phenomenon. Likewise, if you understand DNA, then you will comprehend all biological phenomena. By contrast, a significant aspect of complex systems is that a new collective level of phenomena emerges through interactions between autonomous elements. Hence, in order to comprehend the behavior of a complex system, additional knowledge is required beyond the comprehension of the single elements. More generally, a complex system is structured in a hierarchy of semi-autonomous subsystems, with the behavior of a subsystem at level i of the hierarchy being affected, although not completely determined, by the behavior of each subsystem at level $i-1$. As such, the behavior of a complex system is *stratified*. For example, as outlined in (Kaneko and Tsuda 2013), the brain is structured bottom-up by several spatial subsystems of diverse amplitudes. However, understanding the behavior at one level of the brain hierarchy is not sufficient to understand the behavior at a superior level.

In previous work, starting from (Baroni et al. 1999; Lamperti and Zanella 2000), the authors have addressed model-based diagnosis of a class of discrete-event systems (DESs) (Cassandras and Lafortune 2008), called *active systems* (ASs). Later on, loosely inspired by complexity science, ASs have been aggregated in a hierarchical way, where the behavior of the ASs in each level of the hierarchy is affected by the behavior emerging from the lower level. The interaction between the ASs of two adjacent layers is based on special events, which are generated in the lower layer when specific behavioral patterns occur. This way, a new class of DESs, called *complex active systems* (CASs), built on top of the class of ASs, has been introduced. This paper presents a method to extract knowledge - above all, about emergent behavior - from the models of individual (component) ASs, and to exploit this knowledge for the lazy diagnosis of CASs. *Higher-order DESs*, which are the predecessors of CASs, were first defined in (Lamperti and Zhao 2013b; 2013a). Subsequently, a viable diagnosis technique for higher-order DESs was presented in (Lamperti and Quarenghi 2016; Lamperti and Zhao 2016b) and extended in (Lamperti and Zhao 2016a). However, differently from the method presented in the next sections, this technique neither exploits any compiled knowledge nor processes a single global observation of the system; instead, it assumes that there are several observations, one for each subsystem. To the best of our knowledge, apart from the works cited above, no approach to diagnosis of DESs (much less to diagnosis of static systems) based on the complexity paradigm has been proposed so far. Still, several works can be related to this paper in varying degrees, as discussed in Section 8.

The contribution of this paper is threefold: (a) specification of a CAS based on active units, (b) proposal of a process for extracting knowledge from active units, and (c) specification of a diagnosis task for CASs exploiting compiled knowledge and a totally-ordered temporal observation.

2 Complex Active Systems

A CAS can be defined bottom-up, starting from its atomic elements, the *active components* (ACs). The behavior of an AC, which is equipped with input / output *terminals*, is defined by a communicating automaton (Brand and Zafiropulo 1983). The transition function moves the AC from one state to another when a specific *input event* is *ready* at an input terminal. In so doing, the transition possibly generates a set of *output events* at several output terminals. If specified so, an AC can change its state without the need for a ready event on any input terminal: this is a spontaneous transition, which is typically used for modeling state changes caused by external events (e.g. a lightning may cause the reaction of a sensor in a protection system). ACs communicate with one another through *links*. A link is a connection between an output terminal o of an AC c and an input terminal i' of another AC c'. When an event e is generated at o by a transition in c, e becomes *ready* at terminal i' of c'. At most one link can be connected with a terminal. When several ACs are connected by links, the resulting network is an *active system* (AS) (Lamperti, Zanella, and Zhao 2018). A state of an AS is a pair (S, E), where S is the array of states of the ACs in the AS and E is the array of (possibly empty) events that are ready at the input terminals of the ACs. A state of the AS is *quiescent* iff all elements in E are ε (in practice, no event is ready). A *trajectory* T of an AS is a finite sequence of transitions of ACs in the AS, namely $T = [t_1(c_1), \dots, t_q(c_q)]$, which moves the AS from an *initial* quiescent state to a *final* quiescent state. In an AS, a terminal that is not connected with any link is *dangling*. An *active unit* (AU) is an AS extended by a set of input terminals, a set of output terminals, and a set of *emergent events*, where the input terminals are the dangling input terminals of the AS. Each emergent event is a pair $(e(o), r)$, where e is an event generated at the output terminal o of the AU and r is a regular expression whose alphabet is a subset of the transitions of the ACs involved in the AU. The event $e(o)$ *emerges* from a trajectory of the AS incorporated in the AU when a sequence of transitions in the trajectory matches r. The state of recognition of an emergent event $(e(o), r)$ is maintained by a *matcher*, namely a DFA derived from r, in which each final state corresponds to the occurrence of $e(o)$. A CAS $\mathcal{X}$ is a directed tree, where each node is an AU and each arc $(\mathcal{U}, \mathcal{U}')$, with $\mathcal{U}$ being a child of $\mathcal{U}'$ in the tree, is a set of links connecting all the output terminals of $\mathcal{U}$ with some input terminals of $\mathcal{U}'$. We say that a component / link is in $\mathcal{X}$ if it is in an AU in $\mathcal{X}$.

Example 1 Outlined on the left side of Fig. 1 is a CAS called $\bar{\mathcal{X}}$, involving AUs A, B, and C, where A has one input terminal, B has one input terminal and one output terminal, and C has one output terminal. Since each AU has at most one child, the hierarchy of $\bar{\mathcal{X}}$ is linear (no branches). To avoid irrelevant details, the internal structure of AUs is omitted.

A state of $\mathcal{X}$ is a triple (S, E, M), where S is the array of states of the ACs in $\mathcal{X}$, E is the array of (possibly empty) events ready at the input terminals of the ACs in $\mathcal{X}$, and M is the array of states of the matchers of the events emerging from all AUs in $\mathcal{X}$. Like for ASs, a state of $\mathcal{X}$ is *quiescent* iff all elements in E are ε. A *trajectory* of $\mathcal{X}$ is a finite sequence of transitions of ACs in $\mathcal{X}$, $T = [t_1(c_1), \dots, t_q(c_q)]$, which moves $\mathcal{X}$ from an *initial* quiescent state to a *final* quiescent state. Given an initial state x_0 of $\mathcal{X}$, the *space* of $\mathcal{X}$ is a DFA

$$\mathcal{X}^* = (\Sigma, X, \tau, x_0, X_{\mathrm{f}}), \tag{1}$$

where Σ is the alphabet, namely the set of transitions of ACs in $\mathcal{X}$, X is the set of states, τ is the transition function, namely $\tau : X \times \Sigma \mapsto X$, and $X_{\mathrm{f}} \subseteq X$ is the set of final (quiescent) states. In other words, the sequence of symbols in Σ (transitions of ACs) marking a path (sequence of transitions) in $\mathcal{X}^*$ from x_0 to a final state is a trajectory of $\mathcal{X}$. Hence, $\mathcal{X}^*$ is a finite representation of the (possibly infinite) set of trajectories of $\mathcal{X}$ starting at x_0. Within a trajectory of a CAS $\mathcal{X}$, each transition is either *observable* or *unobservable*. The mode in which an observable transition is perceived is defined by the *viewer* $\mathcal{V}$ of $\mathcal{X}$, namely a set of pairs $(t(c), \ell)$, where $t(c)$ is a transition of an AC c and ℓ is an *observable label*. Given a transition $t(c)$, if $(t(c), \ell) \in \mathcal{V}$, then $t(c)$ is observable via label ℓ; otherwise, $t(c)$ is unobservable. We assume that the sets of observable labels relevant to different AUs is disjoint; in other words, if $(t(c), \ell)$ and $(t(c'), \ell')$ are two pairs in $\mathcal{V}$ such that c and c' belong to different AUs, then $\ell \neq \ell'$. The *temporal observation* $\mathcal{O}$ of a trajectory T of $\mathcal{X}$ based on a viewer $\mathcal{V}$, denoted $T_{\mathcal{V}}$, is the sequence of observable labels associated with the observable transitions in T, namely $\mathcal{O} = [\, \ell \mid t(c) \in T, (t(c), \ell) \in \mathcal{V} \,]$. Not only each transition in a trajectory is either observable or unobservable; it also is either *normal* or *faulty*. Faulty transitions are defined by the *ruler* $\mathcal{R}$ of $\mathcal{X}$, a set of pairs $(t(c), f)$, where $t(c)$ is a transition of an AC c and f is a *fault*. Given a

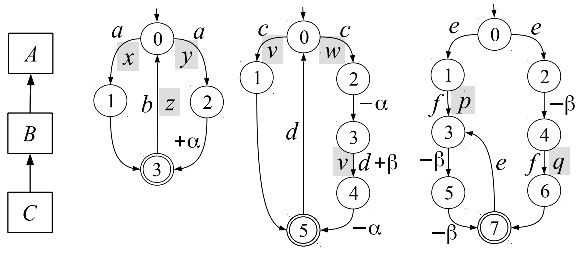
Figure 1: CAS $\bar{\mathcal{X}}$ (left) and unit spaces C^*, B^*, and A^*.

transition $t(c)$, if $(t(c), f) \in \mathcal{R}$, then $t(c)$ is faulty via fault f; otherwise, $t(c)$ is normal. The *diagnosis* of a trajectory T of $\mathcal{X}$ based on $\mathcal{R}$, denoted $T_{\mathcal{R}}$, is the mapping of T to a set of faults defined as[1] $T_{\mathcal{R}} = \{ f \mid t(c) \in T, (t(c), f) \in \mathcal{R} \}$. If $T_{\mathcal{R}} \neq \emptyset$, then T is faulty; otherwise, T is normal. Even if $\mathcal{X}^*$ includes an infinite number of trajectories, the domain of possible diagnoses is always finite, this being the powerset $2^{\mathcal{F}}$, where $\mathcal{F}$ is the domain of the faults involved in $\mathcal{R}$.

3 Diagnosis Problem

When $\mathcal{X}$ performs a trajectory T, what is observed is a temporal observation $\mathcal{O} = T_{\mathcal{V}}$, where $\mathcal{V}$ is the viewer of $\mathcal{X}$. However, owing to partial unobservability, several (possibly infinite) trajectories may be compatible with $\mathcal{O}$. Hence, several *candidate diagnoses* may be compatible with $\mathcal{O}$. The goal is finding all and only these candidates. A *diagnosis problem* $\wp(\mathcal{X}, \mathcal{O})$ is an association between $\mathcal{X}$ and a temporal observation $\mathcal{O}$. Given the viewer $\mathcal{V}$ and the ruler $\mathcal{R}$ for $\mathcal{X}$, the *solution* to $\wp(\mathcal{X}, \mathcal{O})$ is the set of candidate diagnoses

$$\Delta(\wp(\mathcal{X}, \mathcal{O})) = \{ T_{\mathcal{R}} \mid T \in \mathcal{X}^*, T_{\mathcal{V}} = \mathcal{O} \}. \qquad (2)$$

Example 2 Consider the CAS $\bar{\mathcal{X}}$ in Example 1. Assume that: the observable labels involved in its viewer $\bar{\mathcal{V}}$ are a, b, c, d, e, and f; the faults involved in its ruler $\bar{\mathcal{R}}$ are p, q, v, w, x, y, and z; the temporal observation is $\bar{\mathcal{O}} = [a, c, e, d, b, a, f]$. Then, $\wp(\bar{\mathcal{X}}, \bar{\mathcal{O}})$ is a diagnosis problem.

It should be clear that eqn. (2) is only a definition formalizing what the solution to a diagnosis problem is. For self-evident computational complexity reasons, it makes no sense to enumerate the candidate diagnoses as suggested by that definition. The space $\mathcal{X}^*$ plays only a formal role and is never materialized. Even the reconstruction of the subspace of $\mathcal{X}^*$ that is compatible with $\mathcal{O}$ is out of the question because, in the worst case, it suffers from the same computational complexity of $\mathcal{X}^*$. So, what to do? The short answer is: focusing on the AUs rather than on the whole of $\mathcal{X}$. The important points are soundness and completeness: the set of diagnoses determined must coincide with the set of candidate diagnoses defined in eqn. (2).

4 Knowledge Compilation

The diagnosis process involves several tasks, which are performed either *offline* or *online*, that is, *before* or *while* the CAS is being operated, respectively. The tasks performed offline are collectively called "knowledge compilation", and the resulting data structures, "compiled knowledge". Compiled knowledge is meant to speed up the tasks performed online by the diagnosis engine. In offline processing, AUs are processed independently from one another. For each AU $\mathcal{U}$, three data structures are compiled in cascade: the *unit space* $\mathcal{U}^*$, the *unit labeling* $\mathcal{U}_\delta$, and the *unit knowledge* $\mathcal{U}_\Delta$, of which the latter is exploited online. Online processing depends on the diagnosis problem to be solved, specifically, on the observation, whereas offline processing does not.

[1]More generally, any set of faults is called a diagnosis.

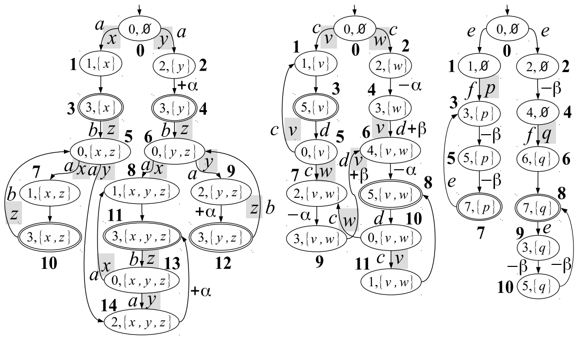
Figure 2: From left to right: unit labelings C_δ, B_δ, A_δ.

Definition 1 (Unit space) *Let $\mathcal{U}$ be an AU involving an AS $\mathcal{A}$. The* space *of $\mathcal{U}$ is a DFA*

$$\mathcal{U}^* = (\Sigma, U, \tau, u_0, U_{\mathrm{f}}), \qquad (3)$$

where: the alphabet Σ is the set of transitions of the ACs in $\mathcal{U}$; U is the set of states (S, E, M), where (S, E) is a state of $\mathcal{A}$ and M is the array of states of the matchers of the emergent events of $\mathcal{U}$; $u_0 = (S_0, E_0, M_0)$ is the initial state, where (S_0, E_0) is a quiescent state of $\mathcal{A}$ and M_0 is the array of the initial states of the matchers; $U_{\mathrm{f}} \subseteq U$ is the set of final states, where $(S, E, M) \in U_{\mathrm{f}}$ iff (S, E) is a quiescent state of $\mathcal{A}$; and $\tau : U \times \Sigma \mapsto U$ is the transition function, where $(S, E, M) \xrightarrow{t(c)} (S', E', M') \in \tau$ iff $(S, E) \xrightarrow{t(c)} (S', E')$ occurs in $\mathcal{A}$, M' is the result of updating M based on $t(c)$, and (S', E', M') is connected to a final state.[2]

Example 3 We start outlining the space of each AU as being generated already, as shown next to the CAS $\bar{\mathcal{X}}$ in Fig. 1, namely C^* (left), B^* (center), and A^* (right). In each unit space, states are identified by numbers, final states are double circled, and transitions are marked by relevant information only, namely: observable label, fault (shaded), occurrence of an emergent event (prefixed by the "+" plus sign), and consumption of an event emerged from a child unit (prefixed by the "−" minus sign). For instance, in B^*, the transition from state 3 to state 4 is observable via label d, is faulty via fault v, and generates the emergent event β. Transition from 2 to 3, which is unobservable and normal, consumes the event α emerging from C, the child of B.

[2]Requiring the connection means that there is a contiguous sequence of transitions from (S', E', M') to a (final) state in U_{f}.

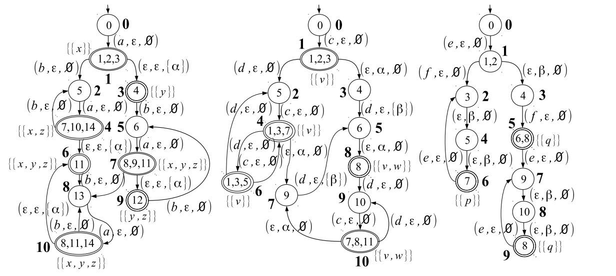

Figure 3: From left to right: unit knowledge C_Δ, B_Δ, A_Δ.

Since the unit space of an AU does not depend on other AUs, the occurrence of a transition depending on an event e emerging from a child AU in the CAS is not constrained by the actual readiness of e at one input terminal, as this information is outside the scope of the AU. Yet, the enforcement of this *interface* constraint is not neglected, but only deferred to the diagnosis engine operating online (cf. Section 5).

Definition 2 (Unit labeling) *Let $\mathcal{U}^* = (\Sigma, U, \tau, u_0, U_\mathrm{f})$ be the space of an AU $\mathcal{U}$ in a CAS $\mathcal{X}$, let $\mathcal{R}$ be the ruler of $\mathcal{X}$, and let $\boldsymbol{\delta}$ be the domain of diagnoses in $\mathcal{U}$. The* labeling *of $\mathcal{U}$ is a DFA*

$$\mathcal{U}_\delta = (\Sigma, U', \tau', u'_0, U'_\mathrm{f}), \tag{4}$$

where: $U' \subseteq U \times \boldsymbol{\delta}$ is the set of states; $u'_0 = (u_0, \emptyset)$ is the initial state; $U'_\mathrm{f} \subseteq U'$ is the set of final states, with $(u, \delta) \in U'_\mathrm{f}$ iff $u \in U_\mathrm{f}$; $\tau' : U' \times \Sigma \mapsto U'$ is the transition function, with $(u, \delta) \xrightarrow{t(c)} (u', \delta') \in \tau'$ iff $u \xrightarrow{t(c)} u' \in \tau$ and

$$\delta' = \begin{cases} \delta \cup \{f\} & \textit{if } (t(c), f) \in \mathcal{R} \\ \delta & \textit{otherwise.} \end{cases}$$

Example 4 Shown in Fig. 2 are the unit labelings C_δ, B_δ, and A_δ derived from the unit spaces in Fig. 1. To facilitate referencing, states are re-identified by numbers. Owing to the additional field δ (set of faults), the number of states in $\mathcal{U}_\delta$ is generally larger than the number of states in $\mathcal{U}^*$.

Definition 3 (Unit knowledge) *Let $\mathcal{U}$ be an AU in a CAS $\mathcal{X}$, let $\mathcal{U}_\delta$ be a labeling of $\mathcal{U}$, and let $\mathcal{V}$ be the viewer of $\mathcal{X}$. Let $\mathcal{U}'_\delta$ be the nondeterministic finite automaton (NFA) obtained from $\mathcal{U}_\delta$ by substituting the alphabet symbols marking the transitions as follows. For each transition $(u, \delta) \xrightarrow{t(c)} (u', \delta')$ in $\mathcal{U}_\delta$, $t(c)$ is replaced with a triple (ℓ, e, E), where: if $(t(c), \ell') \in \mathcal{V}$, then $\ell = \ell'$, else $\ell = \varepsilon$; if $t(c)$ consumes an event e' emerging from a child unit, then $e = e'$, else $e = \varepsilon$; E is the (possibly empty) set of emergent events generated at $t(c)$ by $\mathcal{U}$. Eventually, triples $(\varepsilon, \varepsilon, \emptyset)$ are replaced by ε. The* unit knowledge *$\mathcal{U}_\Delta$ is the DFA obtained by the determinization of $\mathcal{U}'_\delta$, where each final state s_f of $\mathcal{U}_\Delta$ is marked by the aggregation of the diagnosis sets associated with the final states of $\mathcal{U}'_\delta$ within s_f, denoted $\Delta(s_\mathrm{f})$.*[3]

[3]In the *Subset Construction* algorithm (Hopcroft, Motwani, and Ullman 2006), when an NFA is determinized into a DFA, each state in the DFA is identified by a subset of the states of the NFA.

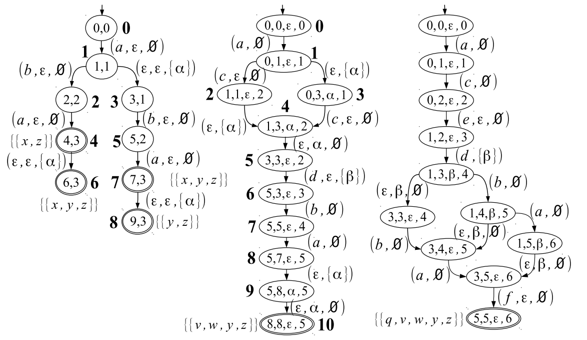

Figure 4: From left to right: unit abductions $C_\mathcal{O}$, $B_\mathcal{O}$, $A_\mathcal{O}$.

Example 5 Shown in Fig. 3 are the unit knowledge C_Δ, B_Δ, and A_Δ, derived from the unit labelings displayed in Fig. 2, where states are re-identified by numbers. Consider the final state 4 of C_Δ (left of Fig. 3), that includes states $7 = (1, \{x, z\})$, $10 = (3, \{x, z\})$, and $14 = (2, \{x, y, z\})$ of C_δ. Since state $10 = (3, \{x, z\})$ in C_δ is final (it is final in C'_δ too), the state 4 of C_Δ is marked by $\{\{x, z\}\}$.

5 Diagnosis Engine

A diagnosis problem for $\mathcal{X}$ is solved online by the *diagnosis engine* exploiting the knowledge of AUs compiled offline. Each unit knowledge $\mathcal{U}_\Delta$ is constrained by the observation of $\mathcal{U}$ and by the events emerging from the child AUs of $\mathcal{U}$.

Definition 4 (Abduction of leaf AU) *Let $\wp(\mathcal{X}, \mathcal{O})$ be a diagnosis problem, let $\mathcal{U}$ be a leaf AU in the tree of $\mathcal{X}$, let $\mathcal{U}_\Delta = (\Sigma, S, \tau, s_0, S_\mathrm{f})$ be the knowledge of $\mathcal{U}$, and let $\mathcal{O}_\mathcal{U} = [\ell_1, \ldots, \ell_{\bar{n}}]$ be the restriction of $\mathcal{O}$ on $\mathcal{U}$, namely*

$$\mathcal{O}_\mathcal{U} = [\, \ell \mid \ell \in \mathcal{O}, (t(c), \ell) \in \mathcal{V}, c \in \mathcal{U} \,]. \tag{5}$$

The abduction *of $\mathcal{U}$ is a DFA*

$$\mathcal{U}_\mathcal{O} = (\Sigma, S', \tau', s'_0, S'_\mathrm{f}),$$

where: $S' \subseteq S \times [0 \mathinner{..} \bar{n}]$ is the set of states[4]*; $s'_0 = (s_0, 0)$ is the initial state; $S'_\mathrm{f} \subseteq S'$ is the set of final states, where $s' \in S'_\mathrm{f}$ iff $s' = (s, \bar{n})$, $s \in S_\mathrm{f}$, with s' being marked by $\Delta(s') = \Delta(s)$; $\tau' : S' \times \Sigma \mapsto S'$ is the transition function, where $(s, j) \xrightarrow{(\ell, \varepsilon, E)} (s', j') \in \tau'$ iff (s', j') is connected to a final state, $s \xrightarrow{(\ell, \varepsilon, E)} s' \in \tau$, and*

$$j' = \begin{cases} j + 1 & \textit{if } \ell \neq \varepsilon \textit{ and } \ell_{j+1} \textit{ (in } \mathcal{O}_\mathcal{U}\textit{)} = \ell \\ j & \textit{if } \ell = \varepsilon. \end{cases} \tag{6}$$

Example 6 Consider the unit knowledge C_Δ on the left side of Fig. 3. Let $\bar{\mathcal{O}}_C = [\, a, b, a \,]$ be the restriction of $\bar{\mathcal{O}}$ on C. The unit abduction $C_\mathcal{O}$ is shown on the left side of Fig. 4.

To extend Def. 4 to nonleaf AUs, we introduce the notion of a unit interface in Def. 5 (generalized in Def. 8).

Definition 5 (Interface of leaf AU) *Let $\mathcal{U}_\mathcal{O}$ be an abduction where $\mathcal{U}$ is a leaf AU. Let $\mathcal{U}'_\mathcal{O}$ be the NFA obtained from $\mathcal{U}_\mathcal{O}$ by substituting (ℓ, E) for each transition symbol*

[4]Each natural number in $[0 \mathinner{..} \bar{n}]$ is called an *index* of $\mathcal{O}$.

(ℓ, ε, E) such that $\ell \neq \varepsilon$ or $E \neq \emptyset$, and ε for all other symbols. The interface *$\mathcal{U}_{\Im}$ of $\mathcal{U}$ is the DFA obtained by determinization of $\mathcal{U}'_{\mathcal{O}}$, where each final state s'_{f} is marked by $\Delta(s'_{\mathrm{f}})$, which is the union of the sets of diagnoses marking the final states of $\mathcal{U}'_{\mathcal{O}}$ included in s'_{f}.*

Example 7 Considering the unit abduction $C_{\mathcal{O}}$ displayed on the left side of Fig. 4, the unit interface $C_{\Im}$ is shown on the left side of Fig. 5. Incidentally, $C_{\Im}$ is isomorphic to $C_{\mathcal{O}}$; hence, states need not to be renamed.

To extend the notion of a unit abduction to nonleaf AUs (Def. 7), we make use of the join operator defined below.

Definition 6 (Join) *The* join *"$\otimes$" of two sets of diagnoses, Δ_1 and Δ_2, is the set of diagnoses defined as follows:*

$$\Delta_1 \otimes \Delta_2 = \{\, \delta \mid \delta = \delta_1 \cup \delta_2, \delta_1 \in \Delta_1, \delta_2 \in \Delta_2 \,\}. \quad (7)$$

Definition 7 (Abduction of nonleaf AU) *Let $\wp(\mathcal{X}, \mathcal{O})$ be a diagnosis problem, let $\mathcal{U}$ be a nonleaf AU in $\mathcal{X}$, let $\bar{\mathcal{U}}$ be the CAS defined by the subtree of $\mathcal{X}$ rooted in $\mathcal{U}$, let $\mathcal{U}_1, \dots, \mathcal{U}_k$ be the child AUs of $\mathcal{U}$ in $\mathcal{X}$, let $\mathcal{U}_\Delta = (\Sigma, S, \tau, s_0, S_{\mathrm{f}})$ be the knowledge of $\mathcal{U}$, let $\mathcal{O}_{\bar{\mathcal{U}}} = [\ell_1, \dots, \ell_{\bar{n}}]$ be the restriction of $\mathcal{O}$ on the CAS $\bar{\mathcal{U}}$, namely*

$$\mathcal{O}_{\bar{\mathcal{U}}} = [\, \ell \mid \ell \in \mathcal{O}, (t(c), \ell) \in \mathcal{V}, c \in \bar{\mathcal{U}} \,], \quad (8)$$

let $\mathbf{E}$ be the domain of tuples of (possibly empty) events emerging from child AUs ready at the input terminals of $\mathcal{U}$, let $\mathcal{U}_{i\Im} = (\Sigma_i, S_i, \tau_i, s_{0i}, S_{\mathrm{f}i})$ be the interface of $\mathcal{U}_i$, $i \in [1 \,..\, k]$, and let $\mathbf{S} = S_1 \times \cdots \times S_k$. The abduction *of $\mathcal{U}$ is a DFA*

$$\mathcal{U}_{\mathcal{O}} = (\Sigma', S', \tau', s'_0, S'_{\mathrm{f}}),$$

where: $\Sigma' = \Sigma \cup \Sigma_1 \cup \cdots \cup \Sigma_k$; $S' \subseteq S \times \mathbf{S} \times \mathbf{E} \times [0 \,..\, \bar{n}]$ is the set of states; $s'_0 = (s_0, (s_{01}, \dots, s_{0k}), (\varepsilon, \dots, \varepsilon), 0)$ is the initial state; $S'_{\mathrm{f}} \subseteq S'$ is the set of final states, where $s'_{\mathrm{f}} \in S'_{\mathrm{f}}$ iff $s'_{\mathrm{f}} = (s_{\mathrm{f}}, (s_{\mathrm{f}1}, \dots, s_{\mathrm{f}k}), (\varepsilon, \dots, \varepsilon), \bar{n})$, $s_{\mathrm{f}} \in S_{\mathrm{f}}$, $\forall i \in [1 \,..\, k]$, $s_{\mathrm{f}i} \in S_{\mathrm{f}i}$, and s'_{f} is marked by the set of diagnoses

$$\Delta(s'_{\mathrm{f}}) = \Delta(s_{\mathrm{f}}) \otimes \Delta(s_{\mathrm{f}1}) \otimes \cdots \otimes \Delta(s_{\mathrm{f}k}); \quad (9)$$

$\tau' : \Sigma' \times S' \mapsto S'$ is the transition function, where

$$(s, (s_1, \dots, s_k), E, j) \xrightarrow{\sigma} (s', (s'_1, \dots, s'_k), E', j') \in \tau'$$

iff $(s', (s'_1, \dots, s'_k), E', j')$ is connected to a final state and either of the following conditions holds:

- *$s \xrightarrow{\sigma} s' \in \tau$, $\sigma = (\ell, e, \bar{E})$, either $e = \varepsilon$ or e is ready in E, E' equals E except that e is removed, and, based on ℓ and $\mathcal{O}_{\bar{\mathcal{U}}}$, the index j' is defined as in eqn. (6);*
- *$s_i \xrightarrow{\sigma} s'_i \in \tau_i$, $\sigma = (\ell, E_i)$, based on ℓ, the index j' is defined as in eqn. (6), and, if $E_i \neq \emptyset$, then all elements in E corresponding to input emergent events in E_i are ε and $E' = E$ except that all events in E_i are inserted into E'.*

Example 8 Consider the unit knowledge B_Δ displayed on the center of Fig. 3 and the unit interface $C_{\Im}$ displayed on the left side of Fig. 5. Let $\bar{\mathcal{O}}_{\bar{B}} = [\, a, c, d, b, a \,]$ be the restriction of $\bar{\mathcal{O}}$ on $\bar{B}$. The unit abduction $B_{\mathcal{O}}$ is shown on the center of Fig. 4. Each state in $B_{\mathcal{O}}$ is marked by a quadruple (s_B, s_C, e, j), where s_B is a state of B_Δ, s_C is a state of $C_{\Im}$ (C is the unique child of B), e is the event emerging

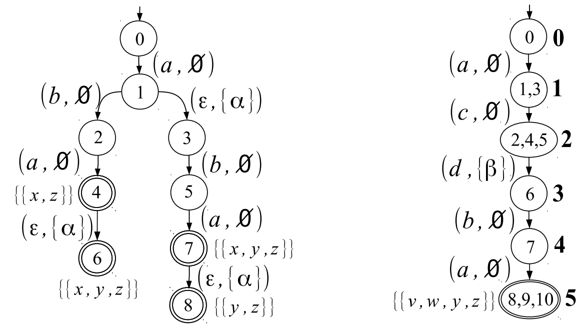

Figure 5: Unit interfaces $C_{\Im}$ (left) and $B_{\Im}$ (right).

from C and possibly ready (if $e \neq \varepsilon$) at the input terminal of B (B includes one input terminal only), and j is an index of the observation $\bar{\mathcal{O}}_{\bar{B}}$. Let $\bar{\Delta}$ be the set of diagnoses marking the final state $10 = (8, 8', \varepsilon, 5)$. Based on eqn. (9), $\bar{\Delta}$ is generated via join $\Delta(8) \otimes \Delta(8')$, where $\Delta(8)$ is the set associated with the state 8 of the unit knowledge B_Δ (shown on the center of Fig. 3), namely $\{\{v, w\}\}$, and $\Delta(8')$ is the set associated with the state 8 of the unit interface $C_{\Im}$ (shown on the left side of Fig. 5), namely $\{\{y, z\}\}$. Hence, $\bar{\Delta} = \{\{v, w\}\} \otimes \{\{y, z\}\} = \{\{v, w, y, z\}\}$. The unit interface $B_{\Im}$, drawn from $B_{\mathcal{O}}$, is displayed on the right side of Fig. 5, where states are renamed.

Definition 8 (Interface of AU) *Let $\mathcal{U}_{\mathcal{O}}$ be an abduction. Let $\mathcal{U}'_{\mathcal{O}}$ be the NFA obtained from $\mathcal{U}_{\mathcal{O}}$ by substituting $(\ell, \emptyset)$ for each transition symbol (ℓ, E) such that $\ell \neq \varepsilon$, then (ℓ, E) for each transition symbol (ℓ, e, E) such that either $\ell \neq \varepsilon$ or $E \neq \emptyset$, and ε for all other symbols. The* interface *$\mathcal{U}_{\Im}$ of $\mathcal{U}$ is the DFA obtained by determinizing $\mathcal{U}'_{\mathcal{O}}$, where each final state s'_{f} is marked by $\Delta(s'_{\mathrm{f}})$, which is the union of the sets of diagnoses marking the final states of $\mathcal{U}'_{\mathcal{O}}$ included in s'_{f}.*

Example 9 Considering the unit abduction $B_{\mathcal{O}}$ displayed on the center of Fig. 4, the unit interface $B_{\Im}$ is shown on the right side of Fig. 5.

6 Soundness and Completeness

Once the abduction process carried out by the diagnosis engine reaches the root $\mathcal{U}$ of the CAS $\mathcal{X}$, thereby generating $\mathcal{U}_{\mathcal{O}}$, the solution to the diagnosis problem $\wp(\mathcal{X}, \mathcal{O})$ can be determined by collecting the diagnoses marking the final states of $\mathcal{U}_{\mathcal{O}}$ (Proposition 1).

Proposition 1 (*Correctness*) *Let $\wp(\mathcal{X}, \mathcal{O})$ be a diagnosis problem, let $\mathcal{U}$ be the AU at the root of $\mathcal{X}$, let $\mathcal{U}_{\mathcal{O}}$ be the abduction of $\mathcal{U}$, with S_{f} being the set of final states, and let*

$$\Delta(\mathcal{U}_{\mathcal{O}}) = \bigcup_{s_{\mathrm{f}} \in S_{\mathrm{f}}} \Delta(s_{\mathrm{f}}) \,. \quad (10)$$

$\Delta(\mathcal{U}_{\mathcal{O}})$ equals the solution $\Delta(\wp(\mathcal{X}, \mathcal{O}))$.

To prove Proposition 1, further terminology is required. Let $\bar{\mathcal{U}}$ be the CAS within $\mathcal{X}$ rooted in $\mathcal{U}$. The *restriction* of $\wp(\mathcal{X}, \mathcal{O})$ on $\bar{\mathcal{U}}$ is a diagnosis problem $\wp(\bar{\mathcal{U}}, \bar{\mathcal{O}})$ where the viewer $\bar{\mathcal{V}}$ of $\bar{\mathcal{U}}$ is the restriction of the viewer of $\mathcal{X}$ on pairs

$(t(c), \ell)$ such that c is a component in $\bar{\mathcal{U}}$, the ruler $\bar{\mathcal{R}}$ of $\bar{\mathcal{U}}$ is the restriction of the ruler of $\mathcal{X}$ on pairs $(t(c), f)$ such that c is a component in $\bar{\mathcal{U}}$, the initial state of $\bar{\mathcal{U}}$ is the restriction of the initial state of $\mathcal{X}$ on the components and links in $\bar{\mathcal{U}}$, and $\bar{\mathcal{O}}$ is the restriction of $\mathcal{O}$ on labels ℓ such that $(t(c), \ell) \in \bar{\mathcal{V}}$.

The notion of a trajectory is extended to any DFA involved in the diagnosis task, namely unit labeling, unit knowledge, unit abduction, and unit interface. A *trajectory* in any such DFA is a string of the regular language of the DFA, that is, the sequence of symbols in the alphabet which mark the sequence of transitions from the initial state to a final state of the DFA; such a final state is called the *accepting state* of the trajectory. The set of trajectories in a DFA $\mathcal{D}$ (the regular language of $\mathcal{D}$) is denoted by $\|\mathcal{D}\|$. For instance, $T \in \|\mathcal{U}_{\mathcal{O}}\|$ denotes a trajectory T in the abduction $\mathcal{U}_{\mathcal{O}}$.

A *path* p in a DFA $\mathcal{D}$ is the sequence of transitions generating a trajectory $[\sigma_1, \ldots, \sigma_n]$ in $\mathcal{D}$, namely $p = s_0 \xrightarrow{\sigma_1} s_1 \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} s_n$. A *semipath* in $\mathcal{D}$ is a prefix of a path in $\mathcal{D}$. A *subpath* p' in $\mathcal{D}$ is a contiguous sequence of transitions within a path, from state s_i to state s_j, namely $s_i \xrightarrow{\sigma_{i+1}} s_{i+1} \xrightarrow{\sigma_{i+2}} \cdots \xrightarrow{\sigma_{j-1}} s_{j-1} \xrightarrow{\sigma_j} s_j$, concisely denoted $s_i \overset{\boldsymbol{\sigma}}{\Rightarrow} s_j$, where $\boldsymbol{\sigma} = [\sigma_{i+1}, \ldots, \sigma_{j-1}, \sigma_j]$ is the *subtrajectory* generated by p'.

The notion of an interface is extended to any trajectory. The *interface* of a trajectory T, denoted $\Im(T)$, is the sequence of pairs (ℓ, E), where ℓ is either an observable label or ε, while E is a (possibly empty) set of emergent events, such that $(\ell, E) \neq (\varepsilon, \emptyset)$. Specifically, if $T \in \|\mathcal{U}_{\Im}\|$, then $\Im(T) = T$. If $T \in \|\mathcal{U}_{\mathcal{O}}\|$, then $\Im(T) = [\,(\ell, E) \mid (\ell, E') \in T, E = \emptyset$ or $(\ell, e, E) \in T\,]$. If $T \in \|\mathcal{U}_{\Delta}\|$, then $\Im(T) = [\,(\ell, E) \mid (\ell, e, E) \in T\,]$. If $T \in \|\mathcal{U}_{\delta}\|$ or $T \in \|\mathcal{U}^*\|$, then, for each $t(c) \in T$, the pair (ℓ, E) in $\Im(T)$ is such that ℓ is either the observable label associated with $t(c)$ or ε, while E is the (possibly empty) set of emergent events at $t(c)$.

Let $\mathcal{G} = (N, A)$ be a directed acyclic graph (DAG), where N is the set of nodes and A is the set of arcs. Let n and n' be two nodes in N. We say that n precedes n', denoted $n \prec n'$, iff n' is reachable from n by a contiguous sequence of arcs. A *topological sort* $\mathcal{S}$ in $\mathcal{G}$ is a sequence of all nodes in N (with each node occurring once) such that, for each pair (n, n') of nodes in $\mathcal{S}$, if $n \prec n'$ in $\mathcal{G}$ then $n \prec n'$ in $\mathcal{S}$. The (finite) set of topological sorts in $\mathcal{G}$ is denoted $\|\mathcal{G}\|$.

Based on the terminology introduced above, a (sketched) proof of Proposition 1 is grounded on Lemma 1.1, Lemma 1.2, Corollary 1.1.1, and Lemma 1.3.

Lemma 1.1 (*Mapping*) *Let $\mathcal{D} = (\Sigma, S, \tau, s_0, S_{\text{f}})$ be a DFA. Let Σ' be an alphabet (possibly including ε) and let $\Sigma \mapsto \Sigma'$ be a mapping from Σ to Σ'. Let $\mathcal{N} = (\Sigma', S, \tau', s_0, S_{\text{f}})$ be the NFA derived from $\mathcal{D}$ by replacing each transition $s \xrightarrow{\sigma} \sigma \in \tau$ with $s \xrightarrow{\sigma'} \sigma \in \tau'$, where σ' is the symbol in Σ' mapping the symbol σ in Σ. Let $\mathcal{D}'$ be the DFA obtained by determinization of $\mathcal{N}$. If T' is a trajectory in $\|\mathcal{D}'\|$ with accepting state s'_{f}, then, for each final state $s_{\text{f}} \in s'_{\text{f}}$, there is a trajectory T in $\|\mathcal{D}\|$ with accepting state s_{f} such that T' is the mapping of T based on $\Sigma \mapsto \Sigma'$.*

Proof. By induction on T'.

(*Basis*) According to the *Subset Construction* determinization algorithm, if $s \in s'_0$, where s'_0 is the initial state of $\mathcal{D}'$, then there is a path $s_0 \overset{\varepsilon^*}{\Rightarrow} s$ in $\mathcal{N}$ where ε^* is a (possibly empty) sequence of ε. Hence, there is a path $s_0 \overset{\boldsymbol{\sigma}}{\Rightarrow} s$ in $\mathcal{D}$ such that ε^* is the mapping of $\boldsymbol{\sigma}$ based on $\Sigma \mapsto \Sigma'$.

(*Induction*) Assume that there is a semipath $s'_0 \overset{\boldsymbol{\sigma}'}{\Rightarrow} s'$ in $\mathcal{D}'$ such that for each $s \in s'$ there is a semipath $s_0 \overset{\boldsymbol{\sigma}}{\Rightarrow} s$ in $\mathcal{D}$ where $\boldsymbol{\sigma}'$ is the mapping of $\boldsymbol{\sigma}$ based on $\Sigma \mapsto \Sigma'$. Consider the new semipath $s'_0 \overset{\boldsymbol{\sigma}'}{\Rightarrow} s' \xrightarrow{\bar{\sigma}} \bar{s}'$ in $\mathcal{D}'$. According to *Subset Construction*, for each state $\bar{s} \in \bar{s}'$ there is a state $s \in s'$ such that $s \overset{\bar{\boldsymbol{\sigma}}_n}{\Longrightarrow} \bar{s}$ is a subpath in $\mathcal{N}$ where $\bar{\boldsymbol{\sigma}}_n$ starts with $\bar{\sigma}$, followed by a (possibly empty) sequence of ε. Hence, there is a subpath $s \overset{\bar{\sigma}_d}{\Longrightarrow} \bar{s}$ in $\mathcal{D}$ such that $\bar{\sigma}_d$ maps to $[\bar{\sigma}]$. Thus, by virtue of the induction hypothesis, $\boldsymbol{\sigma} \cup \bar{\boldsymbol{\sigma}}_d$ maps to $\boldsymbol{\sigma}' \cup [\bar{\sigma}]$ based on $\Sigma \mapsto \Sigma'$. □

Corollary 1.1.1 *With reference to the assumptions stated in Lemma 1.1, if T' is generated by a path $p' = s'_0 \xrightarrow{\sigma'_1} s'_1 \xrightarrow{\sigma'_2} s'_2 \xrightarrow{\sigma'_3} \cdots \xrightarrow{\sigma'_{n-1}} s'_{n-1} \xrightarrow{\sigma'_n} s'_n$ in $\mathcal{D}'$, then there is a path $p = s_0 \overset{\boldsymbol{\sigma}_1}{\Longrightarrow} s_1 \overset{\boldsymbol{\sigma}_2}{\Longrightarrow} s_2 \overset{\boldsymbol{\sigma}_3}{\Longrightarrow} \cdots \overset{\boldsymbol{\sigma}_{n-1}}{\Longrightarrow} s_{n-1} \overset{\boldsymbol{\sigma}_n}{\Longrightarrow} s_n$ in $\mathcal{D}$ such that, $\forall i \in [1 \mathinner{..} n]$, $s_{i-1} \in s'_{i-1}$, $s_i \in s'_i$, and $[\sigma'_i]$ is the mapping of $\boldsymbol{\sigma}_i$ based on $\Sigma \mapsto \Sigma'$.*

Lemma 1.2 (*Soundness*) *If s'_{f} is a final state in $\mathcal{U}_{\mathcal{O}}$, $\delta \in \Delta(s'_{\text{f}})$, and $T \in \|\mathcal{U}_{\mathcal{O}}\|$ with accepting state s'_{f}, then $\delta \in \Delta(\wp(\bar{\mathcal{U}}, \bar{\mathcal{O}})$, where $\delta = \bar{T}_{\bar{\mathcal{R}}}$, $\bar{T} \in \|\bar{\mathcal{U}}^*\|$, and $\Im(\bar{T}) = \Im(T)$.*

Proof. By bottom-up induction on the tree of $\bar{\mathcal{U}}$.

(*Basis*) Assume that $\mathcal{U}$ is a leaf AU. Since $s'_{\text{f}} = (s_{\text{f}}, \bar{n})$ is a final state in $\mathcal{U}_{\mathcal{O}}$, $\delta \in \Delta(s'_{\text{f}})$, and $T \in \|\mathcal{U}_{\mathcal{O}}\|$ with accepting state s'_{f}, based on Def. 4 and Lemma 1.1, there is $T_{\Delta} \in \|\mathcal{U}_{\Delta}\|$ with accepting state s_{f} such that $\delta \in \Delta(s_{\text{f}})$ and $\Im(T_{\Delta}) = \Im(T)$. Hence, based on Def. 3 and Lemma 1.1, there is $T_{\delta} \in \|\mathcal{U}_{\delta}\|$ with accepting state (s, δ) such that $\Im(T_{\delta}) = \Im(T_{\Delta}) = \Im(T)$. Therefore, based on Def. 2, there is $\bar{T} \in \|\mathcal{U}^*\|$ such that $\bar{T}_{\bar{\mathcal{V}}} = \bar{\mathcal{O}}$ and $\bar{T}_{\bar{\mathcal{R}}} = \delta$; in other words, according to eqn. (2), $\delta \in \Delta(\wp(\bar{\mathcal{U}}, \bar{\mathcal{O}})$. Also, $\Im(\bar{T}) = \Im(T_{\delta}) = \Im(T)$.

(*Induction*) Assume that $\mathcal{U}$ is the parent of the AUs $\mathcal{U}_1, \ldots, \mathcal{U}_k$, where $\forall i \in [1 \mathinner{..} k]$, if $s'_{i\text{f}}$ is a final state in $\mathcal{U}_{i\mathcal{O}}$, $\delta_i \in \Delta(s'_{i\text{f}})$, and $T_i \in \|\mathcal{U}_{i\mathcal{O}}\|$ with accepting state $s'_{i\text{f}}$, then $\delta_i \in \Delta(\wp(\bar{\mathcal{U}}_i, \bar{\mathcal{O}}_i)$, where $\delta_i = \bar{T}_{i\bar{\mathcal{R}}_i}$, $\bar{T}_i \in \|\bar{\mathcal{U}}^*_i\|$, and $\Im(\bar{T}_i) = \Im(T_i)$.

Since $s'_{\text{f}} = (s_{\text{f}}, (s_{1\text{f}}, \ldots, s_{k\text{f}}), (\varepsilon, \ldots, \varepsilon), n)$ is a final state in $\mathcal{U}_{\mathcal{O}}$, $\delta \in \Delta(s'_{\text{f}})$, and $T \in \|\mathcal{U}_{\mathcal{O}}\|$ with accepting state s'_{f}, according to eqn. (9), $\delta \in \Delta(s'_{\text{f}}) = \Delta(s_{\text{f}}) \otimes \Delta(s_{1\text{f}}) \otimes \cdots \otimes \Delta(s_{k\text{f}})$, where, $\forall i \in [1 \mathinner{..} k]$, $s_{i\text{f}}$ is final in $\mathcal{U}_{i\Im}$. Hence, based on Def. 6, $\delta = \delta_0 \cup \delta_1 \cup \cdots \cup \delta_k$, where $\delta_0 \in \Delta(s_{\text{f}})$ and, $\forall i \in [1 \mathinner{..} k]$, $\delta_i \in \Delta(s_{i\text{f}})$. Also, based on Def. 8 and Lemma 1.1, $\forall i \in [1 \mathinner{..} k]$, there is $T_i \in \|\mathcal{U}_{i\mathcal{O}}\|$ with accepting state $s'_{i\text{f}} \in s_{i\text{f}}$ such that $\delta_i \in s'_{i\text{f}}$, all emergent events involved in T_i are consumed in T, and all the (nonempty) observable labels involved in T_i are also involved in T with the same relative order. Thus, by virtue of the induction hypothesis, we have $\delta_i \in \Delta(\wp(\bar{\mathcal{U}}_i, \bar{\mathcal{O}}_i)$, where $\delta_i = \bar{T}_{i\bar{\mathcal{R}}_i}$, $\bar{T}_i \in \|\bar{\mathcal{U}}^*_i\|$, and $\Im(\bar{T}_i) = \Im(T_i)$. According to Def. 7, the trajectory T is composed of triples (ℓ, e, E) interspersed by pairs (ℓ_i, E_i), with each nonempty ℓ_i being an observable label and each nonempty E_i being a set of events emerging from $\mathcal{U}_i$. Note that, for each (ℓ_i, E_i) in T there is one transition in $\bar{T}_i$ generating ℓ_i and E_i. Thus, each $\bar{T}_i$ is composed of transitions $t(c)$, where c is a component in $\bar{\mathcal{U}}_i$, in which there are some transitions generating ℓ_i and E_i. The sequence of (ℓ_i, E_i) generated in $\bar{T}_i$ equals the subsequence of (ℓ_i, E_i) in T, $i \in [1 \mathinner{..} k]$. So, there is a sequential correspondence between each (ℓ_i, E_i) in T and each transition in $\bar{T}_i$ generating ℓ_i and E_i, namely (ℓ_i, E_i). According to Def. 7 and Corollary 1.1.1, if $s_0' \overset{\boldsymbol{\sigma}_1'}{\Longrightarrow} s_1' \xrightarrow{(\ell_{i1},E_{i1})} s_1'' \overset{\boldsymbol{\sigma}_2'}{\Longrightarrow} s_2' \xrightarrow{(\ell_{i2},E_{i2})} s_2'' \overset{\boldsymbol{\sigma}_3'}{\Longrightarrow} \cdots \overset{\boldsymbol{\sigma}_n'}{\Longrightarrow} s_n'$ is the path in $\mathcal{U}_{\mathcal{O}}$ generating the trajectory $T = \boldsymbol{\sigma}_1' \cup [(\ell_{i1}, E_{i1})] \cup \boldsymbol{\sigma}_2' \cup [(\ell_{i2}, E_{i2})] \cup \boldsymbol{\sigma}_3' \cup \cdots \cup \boldsymbol{\sigma}_n'$, then there is a path $s_0 \overset{\boldsymbol{\sigma_1}}{\Longrightarrow} s_1 \overset{\boldsymbol{\sigma_2}}{\Longrightarrow} s_2 \overset{\boldsymbol{\sigma_3}}{\Longrightarrow} \cdots \overset{\boldsymbol{\sigma_n}}{\Longrightarrow} s_n$ in $\mathcal{U}_\delta$ generating the trajectory $T^* = \boldsymbol{\sigma_1} \cup \boldsymbol{\sigma_2} \cup \cdots \cup \boldsymbol{\sigma_n}$, $T^* \in \|\mathcal{U}^*\|$, such that T^* generates the restriction of $\bar{\mathcal{O}}$ on $\mathcal{U}$ and the diagnosis δ_0. Now, consider the DAG $\mathcal{G}$ constructed by the following four steps, where each node is either a component transitions or a pair (ℓ_i, E_i), while arcs indicate precedence dependencies between these nodes. Step 1: in T, substitute $\boldsymbol{\sigma}_j$ for each $\boldsymbol{\sigma}_j'$, $j \in [1 \mathinner{..} n]$; this step substitutes sequences of transitions of components in $\mathcal{U}$ for corresponding sequences of triples (ℓ, e, E) in T. Step 2: transform each trajectory $\bar{T}_i = [t_1, t_2, \ldots, t_{n_i}]$, $i \in [1 \mathinner{..} k]$, into a connected sequence of nodes $t_1 \to t_2 \to \cdots \to t_{n_i}$, where arrows indicate arcs (precedence dependencies). Step 3: transform T into a connected sequence of nodes in the same way as in step 2. Step 4: for each transition t_i in $\bar{T}_i$ generating (ℓ_i, E_i), insert an arc from t_i to the corresponding (ℓ_i, E_i) in T. This results in a DAG, namely a *dependency graph* $\mathcal{G}$, where each (ℓ_i, E_i) is entered by an arc from a transition in $\bar{T}_i$, $i \in [1 \mathinner{..} k]$. Let $\bar{T}$ be the sequence of transitions relevant to a topological sort of $\mathcal{G}$, where pairs (ℓ_i, E_i) are eventually removed. As such, $\bar{T}$ is a sequence of transitions of components in $\bar{\mathcal{U}}$ fulfilling the precedences imposed by $\mathcal{G}$. Remarkably, $\bar{T}$ fulfills the following properties: (1) $\bar{T} \in \|\bar{\mathcal{U}}^*\|$, because the component transitions in each $\bar{T}_i$ are only constrained by the emptiness of the output terminals of $\mathcal{U}_i$, while the component transitions in T are only constrained by the availability of the events emerging from $\mathcal{U}_1, \ldots, \mathcal{U}_k$, which are checked by the mode in which the abduction $\mathcal{U}_{\mathcal{O}}$ is generated; (2) $\bar{T}_{\bar{\mathcal{V}}} = \bar{\mathcal{O}}$, because the sequence of observable labels generated by transitions in $\bar{T}$ equals $\bar{\mathcal{O}}$; and (3) the sequence of faults generated by the transitions in T equals δ_0 and, $\forall i \in [1 \mathinner{..} k]$, $\bar{T}_{i\bar{\mathcal{R}}_i} = \delta_i$. In other words, $\bar{T}_{\bar{\mathcal{R}}} = \delta = \delta_0 \cup \delta_1 \cup \cdots \cup \delta_k$; hence, $\delta \in \Delta(\wp(\bar{\mathcal{U}}, \bar{\mathcal{O}})$. Also, $\Im(\bar{T}) = \Im(T)$. □

Lemma 1.3 (*Completeness*) *If* $\delta \in \Delta(\wp(\bar{\mathcal{U}}, \bar{\mathcal{O}})$*, where* $\delta = \bar{T}_{\bar{\mathcal{R}}}$ *and* $\bar{T} \in \|\bar{\mathcal{U}}^*\|$*, then there is* $T \in \|\mathcal{U}_{\mathcal{O}}\|$ *ending in* s_{f}' *such that* $\delta \in \Delta(s_{\mathrm{f}}')$ *and* $\Im(T) = \Im(\bar{T})$*.*

Proof. By bottom-up induction on the tree of $\bar{\mathcal{U}}$.

(*Basis*) Assume that $\mathcal{U}$ is a leaf AU. As $\delta \in \Delta(\wp(\bar{\mathcal{U}}, \bar{\mathcal{O}})$, where $\delta = \bar{T}_{\bar{\mathcal{R}}}$ and $\bar{T} \in \|\bar{\mathcal{U}}^*\|$, based on Def. 2, there is $T_\delta \in \|\mathcal{U}_\delta\|$ with accepting state (s, δ) such that $\Im(T_\delta) = \Im(\bar{T})$. Hence, based on Def. 3, there is $T_\Delta \in \|\mathcal{U}_\Delta\|$ with accepting state s_{f} such that $\delta \in \Delta(s_{\mathrm{f}})$ and $\Im(T_\Delta) = \Im(T_\delta) = \Im(\bar{T})$. Thus, based on Def. 4, there is $T \in \|\mathcal{U}_{\mathcal{O}}\|$ ending in $s_{\mathrm{f}}' = (s_{\mathrm{f}}, n)$ such that $\delta \in \Delta(s_{\mathrm{f}}')$ and $\Im(T) = \Im(T_\Delta) = \Im(\bar{T})$.

(*Induction*) Assume that $\mathcal{U}$ is the parent of $\mathcal{U}_1, \ldots, \mathcal{U}_k$, where, $\forall i \in [1 \mathinner{..} k]$, if $\delta_i \in \Delta(\wp(\bar{\mathcal{U}}_i, \bar{\mathcal{O}}_i))$, where $\delta_i = \bar{T}_{i\bar{\mathcal{R}}_i}$ and $\bar{T}_i \in \|\bar{\mathcal{U}}_i^*\|$, then there is $T_i \in \|\mathcal{U}_{i\mathcal{O}}\|$ with accepting state $s_{i\mathrm{f}}'$ such that $\delta_i \in \Delta(s_{i\mathrm{f}}')$ and $\Im(T_i) = \Im(\bar{T}_i)$.

Since $\delta \in \Delta(\wp(\bar{\mathcal{U}}, \bar{\mathcal{O}})$, where $\delta = \bar{T}_{\bar{\mathcal{R}}}$ and $\bar{T} \in \|\bar{\mathcal{U}}^*\|$, we have $\delta = \delta_0 \cup \delta_1 \cup \cdots \cup \delta_k$, where δ_0 includes the faults of the components in the AU $\mathcal{U}$ and, $\forall i \in [1 \mathinner{..} k]$, δ_i includes the faults of the components in the CAS $\bar{\mathcal{U}}_i$. Since each δ_i is the diagnosis of the trajectory $\bar{T}_i$ corresponding to the restriction of $\bar{T}$ on the transitions of the components in $\bar{\mathcal{U}}_i$ such that $\bar{T}_i \in \|\bar{\mathcal{U}}_i^*\|$, $\bar{T}_{i\bar{\mathcal{V}}_i} = \bar{\mathcal{O}}_i$, and $\bar{T}_{i\bar{\mathcal{R}}_i} = \delta_i$, based on eqn. (2), $\delta_i \in \wp(\bar{\mathcal{U}}_i, \bar{\mathcal{O}}_i)$. Hence, by virtue of the induction hypothesis, there is $T_i \in \|\mathcal{U}_{i\mathcal{O}}\|$ with accepting state $s_{i\mathrm{f}}'$ such that $\delta_i \in \Delta(s_{i\mathrm{f}}')$ and $\Im(T_i) = \Im(\bar{T}_i)$. Also, based on Def. 8, there is $T_i' \in \|\mathcal{U}_{i\Im}\|$ with accepting state $s_{i\mathrm{f}}$ such that $\delta_i \in \Delta(s_{i\mathrm{f}})$ and $\Im(T_i') = \Im(T_i) = \Im(\bar{T}_i)$. Let T' be the subtrajectory of $\bar{T}$ including only the transitions of components in $\mathcal{U}$. As such, $T' \in \|\mathcal{U}^*\|$, $T_{\bar{\mathcal{V}}}'$ equals the restriction of $\bar{\mathcal{O}}$ on $\mathcal{U}$, and $T_{\bar{\mathcal{R}}}' = \delta_0$. Hence, based on Def. 2, there is $T_\delta \in \|\mathcal{U}_\delta\|$ with accepting state (s, δ_0) such that $\Im(T_\delta) = \Im(T')$. Also, based on Def. 3, there is $T_\Delta \in \|\mathcal{U}_\Delta\|$ with accepting state s_{f} such that $\delta \in \Delta(s_{\mathrm{f}})$ and $\Im(T_\Delta) = \Im(T_\delta) = \Im(T')$. Thus, based on Def. 7, there is $T \in \|\mathcal{U}_{\mathcal{O}}\|$ with accepting state $s_{\mathrm{f}}' = (s_{\mathrm{f}}, (s_{1\mathrm{f}}, \ldots, s_{k\mathrm{f}}), (\varepsilon, \ldots, \varepsilon), n)$ such that $\delta_0 \in \Delta(s_{\mathrm{f}})$ and, $\forall i \in [1 \mathinner{..} k]$, $\delta_i \in \Delta(s_{i\mathrm{f}})$. Since $\delta = \delta_0 \cup \delta_1 \cup \cdots \cup \delta_k$, based on Def. 6 and according to eqn. (9), $\delta \in \Delta(s_{\mathrm{f}}) \otimes \Delta(s_{1\mathrm{f}}) \otimes \cdots \otimes \Delta(s_{k\mathrm{f}})$; that is, $\delta \in \Delta(s_{\mathrm{f}}')$. Also, $\Im(T) = \Im(\bar{T})$. □

Example 10 Consider the unit knowledge A_Δ on the right side of Fig. 3 and the unit interface $B_\Im$ on the right side of Fig. 5. We have $\bar{\mathcal{O}}_{\bar{A}} = \mathcal{O} = [a, c, e, d, b, a, f]$. The unit abduction $A_{\mathcal{O}}$ is shown on the right side of Fig. 4. Let $\bar{\Delta}$ be the set of diagnoses marking the final state $(5, 5', \varepsilon, 6)$. Based on eqn. (9), $\bar{\Delta}$ is generated via join $\Delta(5) \otimes \Delta(5')$, where $\Delta(5)$ is the set associated with the state 5 of the unit knowledge A_Δ (on the right side of Fig. 3), namely $\{\{q\}\}$, and $\Delta(5')$ is the set associated with the state 5 of the unit interface $B_\Im$ (on the right side of Fig. 5), namely $\{\{v, w, y, z\}\}$. Hence, $\bar{\Delta} = \{\{q, v, w, y, z\}\}$, which, incidentally, is a singleton. Based on Proposition 1, $\bar{\Delta}$ is the solution to the diagnosis problem $\wp(\bar{\mathcal{X}}, \bar{\mathcal{O}})$ defined in Example 2.

7 Computational Complexity

If we had adapted the diagnosis technique proposed for ASs (Lamperti, Zanella, and Zhao 2018) to the diagnosis of CASs, then we would have inherited (and even exacerbated) its poor performance, as shown by the experimental comparison in (Lamperti and Quarenghi 2016). In contrast with diagnosis of ASs, the diagnosis engine described in Section 5 does not require the abduction of the whole system. Instead, it focuses on the abduction of each single AU based on the temporal observation and the interface constraints coming from the child AUs. In doing so, it exploits the unit knowledge compiled offline.

We analyze the time complexity of solving a diagnosis problem $\wp(\mathcal{X}, \mathcal{O})$ based on the (not unreasonable) assumption that the processing time is proportional to the size (number of states) of the data structures generated by the diagnosis engine. Furthermore, we make the following *bounding assumptions*: $\mathcal{X}$ is composed of n AUs; each nonleaf AU has c child AUs, has h input terminals, and defines m emergent events; each unit knowledge (generated offline) includes k states; the length of the temporal observation $\mathcal{O}$ is o. We also assume that the size of the DFA obtained by the determinization of an NFA compares to the size of the NFA.[5] We call this the *determinization assumption*. We first consider the (upper bound of the) complexity $\mathcal{C}$ of the abduction $\mathcal{U}_{\mathcal{O}}$ of a leaf AU (at level zero, that is, without children). Based on Def. 4, $\mathcal{C}_{\mathcal{U}_{\mathcal{O}}} = k \cdot o$. In order to estimate the complexity of each unit abduction $\mathcal{U}_{\mathcal{O}}$ at level one, where all children of $\mathcal{U}$ are leaf AUs, two steps have to be analyzed: (1) generation of c interfaces, and (2) generation of $\mathcal{U}_{\mathcal{O}}$ based on Def. 7. As to step (1), on the ground of the determinization assumption, the number of states of the interfaces of the child AUs is $c \cdot k \cdot o$. Based on Def. 7, the (upper bound of the) number of states generated by step (2) for each unit abduction at level one is $k \cdot (k \cdot o)^c \cdot m^h \cdot o = (k \cdot o)^{c+1} \cdot m^h$. Owing to the factor $(k \cdot o)^{c+1}$, the contribution $c \cdot k \cdot o$ of step (1) can be neglected; hence, $\mathcal{C}_{\mathcal{U}_{\mathcal{O}}} = (k \cdot o)^{c+1} \cdot m^h$. The complexity of each unit abduction $\mathcal{U}_{\mathcal{O}}$ at the second level (where $\mathcal{U}$ is the grandparent of some leaf AUs) can be computed as before, where the size of each interface equals $\mathcal{C}_{\mathcal{U}_{\mathcal{O}}}$, namely

$$\mathcal{C}_{\mathcal{U}_{\mathcal{O}}} = k \cdot ((k \cdot o)^{c+1} \cdot m^h)^c \cdot m^h \cdot o = (k \cdot o)^{c^2+c+1} \cdot m^{(c+1) \cdot h}$$

At level d (depth of the tree) of the root AU, the complexity of the unit abduction is

$$\mathcal{C}_{\mathcal{U}_{\mathcal{O}}} = (k \cdot o)^{c^d + c^{d-1} + \cdots + c^2 + c + 1} \cdot m^{(c^{d-1} + \cdots + c^2 + c + 1) \cdot h}$$

As $1 + c + c^2 + \cdots + c^d = n$, where n is the number of AUs in the CAS, the dominant factor in the equation above is $(k \cdot o)^n$. In other words, the complexity of the unit abduction is exponential in the number of AUs in the CAS.

Finally, we consider the space complexity of knowledge compilation, which is performed offline. We assume that each AU includes p components and l links, with each component having s states and generating q different events for each connected link. Each matcher (the DFA recognizing the occurrence of an emergent event) has μ states. The number of possible faults is f. The space complexity of the unit space $\mathcal{U}^*$ is $\mathcal{C}_{\mathcal{U}^*} = s^p \cdot (q+1)^\ell \cdot \mu^m$. The space complexity of the unit labeling $\mathcal{U}_\delta$ is $\mathcal{C}_{\mathcal{U}_\delta} = \mathcal{C}_{\mathcal{U}^*} \cdot 2^f = s^p \cdot (q+1)^\ell \cdot \mu^m \cdot 2^f$. According to the determinization assumption, the complexity of the unit knowledge equals the complexity of the unit labeling, namely $\mathcal{C}_{\mathcal{U}_\Delta} = \mathcal{C}_{\mathcal{U}_\delta}$.

[5]In the worst case, if n is the number of states of the NFA, then the number of states of the DFA is 2^n. However, this is an extremely improbable scenario since, in practice, the size of the DFA resulting from the determinization of an NFA generated randomly typically compares with the size of the NFA (cf. Fig. 3). If the NFA includes a considerable percentage of ε-transitions, then the size of the DFA is likely to be substantially smaller than the size of the NFA (Brognoli, Lamperti, and Scandale 2016).

8 Related Work

Enhancing the online performances of DES diagnosis by exploiting some compiled knowledge produced offline is not a novelty. The seminal work (Sampath et al. 1995) on diagnosability and diagnosis of DESs proposed to draw offline a so-called *diagnoser*, relevant to the whole DES, from the models of its components, and to use it online, thus processing the observation relevant to a diagnosis problem instance in a time that is linear in the length of the observation. Still, differently from CASs, the DESs addressed by (Sampath et al. 1995) are flat. Moreover, the method to generate the diagnoser is not viable, as it needs the one-shot construction of the behavioral model of the whole DES, which is computationally prohibitive. The method described in this paper, instead, trades off some online efficiency for the feasibility of offline processing. In fact, neither the global model nor the diagnoser of the whole system is produced; instead, the knowledge compiled offline is distributed into different units that could possibly be processed in parallel. Online, when the system observation is available, each knowledge unit is first processed separately, and then the results of the abductive reasoning are progressively combined.

Each event emerging from a layer of a CAS hierarchy, directed to the adjacent upper layer, corresponds to a regular expression in the language of the state transitions. In other words, an emergent event is generated when a behavioral pattern is recognized (i.e., the final state of a DFA is reached). The notion of a behavioral pattern is similar to that of *fault supervision pattern*, defined in (Jéron et al. 2006). However, the two categories of patterns are orthogonal: supervision patterns are meant to generalize the concept of fault for both diagnosis and diagnosability of flat DESs, while patterns of emergent events are aimed at supporting behavior stratification in CASs.

CASs are endowed with a tree-shaped hierarchical structure similar to that of hierarchical finite state machines (HFSMs), which are inspired by state-charts (Harel 1987). Diagnosis of HFSMs was considered in (Idghamishi and Zad 2004; Paoli and Lafortune 2008). However, the complexity of HFSMs is confined to structure, without emergent events or behavior stratification, as instead is the case with CASs.

Similarly, no emergent behavior is conceived in (Stumptner and Wotawa 2001), where an algorithm for computing minimal diagnoses of tree-structured systems is presented. The goal is to improve the efficiency of the diagnosis task by exploiting the structure of the system. Subdiagnoses, which are generated by traversing the tree top-down, are eventually combined to yield the candidate diagnoses of the whole system. The considered systems are static while CASs are dynamic, and the diagnosis method is consistency-based, whereas the method described in this paper for CAS diagnosis is abductive and processes the tree structure bottom-up.

An approach to consistency-based diagnosis supported by structural abstraction (which aggregates components to represent a static system at different levels of structural detail) is described in (Chittaro and Ranon 2004) as a remedy for the high computational complexity of model-based diagnosis. Evidence from experimental evaluation indicates that, on average, the technique performs favorably with the algorithm in (Mozetič 1992). Differently from the topic of this paper, the approach is consistency-based, the considered systems are static and without emergent behavior.

A hierarchical technique for consistency-based diagnosis of multiple faults in combinational circuits is presented in (Siddiqi and Huang 2007). Its effectiveness has been demonstrated by an implementation on top of the tool described in (Huang and Darwiche 2005), which assumes that the system model has been compiled into propositional formulas in decomposable negation normal form (DNNF). However, differently from CASs, the systems addressed are static and their models encompass neither emergent behavior nor behavior stratification.

In contrast with the technique for diagnosability checking of DESs proposed in (Sampath et al. 1995), which suffers from exponential complexity, the *twin plant* method (Jiang et al. 2001; Yoo and Lafortune 2002) exhibits a polynomial complexity. However, the construction of a global twin plant, which corresponds to the synchronization based on (all) observable events of two identical instances of the automaton representing the whole DES behavior, is often impractical. The scalability of the approach is increased in (Schumann and Huang 2008) by taking advantage of the distribution of a DES to build a local twin plant for each component. Then, the DES components (and their relevant local twin plants) are organized into a *jointree*, a classical tool adopted in various fields of AI, including probabilistic reasoning (Shenoy and Shafer 1986; Huang and Darwiche 1996) and constraint processing (Dechter 2003). Both the method in (Schumann and Huang 2008) and that presented in this paper are distributed. However, the tasks accomplished by the two methods are different, and the transformation of the DES into a jointree in (Schumann and Huang 2008) is an artifice for reducing the complexity of the diagnosability analysis while the tree-shaped structure is an integral part of the topological and behavioral abstraction in CAS modeling.

A decentralized / distributed approach to diagnosis of DESs is introduced in (Kan John and Grastien 2008), with the aim of computing local diagnoses that are globally consistent. The DES is transformed into a jointree (called *junction tree* in the paper) in order to mitigate the complexity of model-based diagnosis of DESs associated with abduction-based elicitation of system trajectories. Although the tasks accomplished by the methods in (Kan John and Grastien 2008) and in the current paper are the same and both approaches are distributed, they differ in several respects, first of all for in (Kan John and Grastien 2008) the diagnosis task is performed online only, without exploiting any compiled knowledge, and the considered DESs are plain, with no emergent behavior.

9 Conclusion

The contributions of this paper are the specification of a class of DESs inspired by the complexity paradigm and a knowledge-compilation technique that speeds up the online diagnosis. The shift from ASs to CASs does not come with any additional cost in the diagnosis task, which is not only sound and complete, but also viable compared to the diagnosis of ASs. In fact, since a state of the AS includes the states of *all* components and the states of *all* links, the complexity of the abduction of the whole AS in diagnosis of ASs is exponential both in the number of components *and* in the number of links. This theoretical expectation is confirmed by the experimental results presented in (Lamperti and Quarenghi 2016), with the diagnosis engine being not supported by any compiled knowledge. Two diagnosis engines have been implemented, one *greedy* and one *lazy*. The greedy engine makes use of the same technique of behavior reconstruction adopted in diagnosis of ASs (Lamperti, Zanella, and Zhao 2018), while the lazy engine operates similarly to the technique proposed in this paper (although without any compiled knowledge). The results clearly show that the processing time of the lazy engine increases almost linearly with the size of the system, in contrast with the processing time of the greedy engine, which grows exponentially. On the ground of these results, we expect that the technique proposed in this paper, which is essentially a lazy engine exploiting compiled knowledge, is still more efficient than the lazy engine in (Lamperti and Quarenghi 2016), since the low-level model-based reasoning, performed offline and incorporated in the compiled knowledge, is avoided online. Experiments to confirm this intuition will be carried out.

But, why should we model a real system as a CAS rather than a flat AS? In our opinion, the reason is twofold. First, several real event-driven systems are typically organized hierarchically, at different abstraction levels. Modeling one such system as an AS may be awkward because of the mismatch between the flat organization of the model and the hierarchical structure and behavior of the real system. In short, the first benefit is ergonomics in the modeling task. Second, once the real system has been modeled as a CAS, the diagnosis task is viable and provides the sound and complete solution more efficiently than in a (flat) AS.

Several research paths can be envisaged. First, the tree-based topology of the CAS can be relaxed to a directed acyclic graph (DAG), where an AU can have several parent units. Second, the process of knowledge compilation can be extended in order to generate a fast *diagnoser* of the CAS, that is, a DFA whose language includes all possible temporal observations of the CAS. This way, the accepting state of a given temporal observation is marked by the solution to the corresponding diagnosis problem. Hence, the online diagnosis task can be performed in a time that is linear in the length of the temporal observation. Third, the diagnosis task, which in this paper is assumed to be *a posteriori*, that is, carried out at the reception of a complete temporal observation of the CAS, can be made *reactive*, where diagnosis is performed as soon as a piece of observation is received. Fourth, the language of the patterns defining emergent events can be extended beyond regular expressions, based on more powerful grammars. Finally, in order to avoid the generation of the whole set of candidate diagnoses, the search space might be pruned based on specified criteria.

Acknowledgments

This work was supported in part by Regione Lombardia (Smart4CPPS, Linea Accordi per Ricerca, Sviluppo e Innovazione, POR-FESR 2014-2020 Asse I).

References

Atay, F., and Jost, J. 2004. On the emergence of complex systems on the basis of the coordination of complex behaviors of their elements: synchronization and complexity. *Complexity* 10(1):17–22.

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

Bossomaier, T., and Green, D. 2007. *Complex Systems*. Cambridge University Press.

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

Brognoli, S.; Lamperti, G.; and Scandale, M. 2016. Incremental determinization of expanding automata. *The Computer Journal* 59(12):1872–1899.

Cassandras, C., and Lafortune, S. 2008. *Introduction to Discrete Event Systems*. New York: Springer Science+Business Media, second edition.

Chittaro, L., and Ranon, R. 2004. Hierarchical model-based diagnosis based on structural abstraction. *Artificial Intelligence* 155(1–2):147–182.

Dechter, R. 2003. *Constraint Processing*. The Morgan Kaufmann Series in Artificial Intelligence. San Francisco, CA: Morgan Kaufmann Publishers Inc.

Goles, E., and Martinez, S., eds. 2001. *Complex Systems*. Dordrecht, Netherlands: Springer Science+Business Media.

Harel, D. 1987. Statecharts: a visual formalism for complex systems. *Science of Computer Programming* 8(3):231–274.

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

Huang, C., and Darwiche, A. 1996. Inference in belief networks: A procedural guide. *International Journal of Approximate Reasoning* 15(3):225–263.

Huang, J., and Darwiche, A. 2005. On compiling system models for faster and more scalable diagnosis. In *20th National Conference on Artificial Intelligence (AAAI'05)*, 300–306.

Idghamishi, A., and Zad, S. 2004. Fault diagnosis in hierarchical discrete-event systems. In *43rd IEEE Conference on Decision and Control*, 63–68.

Jéron, T.; Marchand, H.; Pinchinat, S.; and Cordier, M. 2006. Supervision patterns in discrete event systems diagnosis. In *Seventeenth International Workshop on Principles of Diagnosis (DX'06)*, 117–124.

Jiang, S.; Huang, Z.; Chandra, V.; and Kumar, R. 2001. A polynomial algorithm for testing diagnosability of discrete event systems. *IEEE Transactions on Automatic Control* 46(8):1318–1321.

Kan John, P., and Grastien, A. 2008. Local consistency and junction tree for diagnosis of discrete-event systems. In *Eighteenth European Conference on Artificial Intelligence (ECAI 2008)*, 209–213. Patras, Greece: IOS Press, Amsterdam.

Kaneko, K., and Tsuda, I. 2013. *Complex Systems: Chaos and Beyond: a Constructive Approach with Applications in Life*. Springer, Berlin, Heidelberg.

Lamperti, G., and Quarenghi, G. 2016. Intelligent monitoring of complex discrete-event systems. In Czarnowski, I.; Caballero, A.; Howlett, R.; and Jain, L., eds., *Intelligent Decision Technologies 2016*, volume 56 of *Smart Innovation, Systems and Technologies*. Springer International Publishing Switzerland. 215–229.

Lamperti, G., and Zanella, M. 2000. Generation of diagnostic knowledge by discrete-event model compilation. In *Seventh International Conference on Knowledge Representation and Reasoning (KR 2000)*, 333–344.

Lamperti, G., and Zhao, X. 2013a. Diagnosis of higher-order discrete-event systems. In Cuzzocrea, A.; Kittl, C.; Simos, D.; Weippl, E.; and Xu, L., eds., *Availability, Reliability, and Security in Information Systems and HCI*, volume 8127 of *Lecture Notes in Computer Science*. Berlin, Heidelberg: Springer. 162–177.

Lamperti, G., and Zhao, X. 2013b. Specification and model-based diagnosis of higher-order discrete-event systems. In *IEEE International Conference on Systems, Man, and Cybernetics (SMC 2013)*, 2342–2347.

Lamperti, G., and Zhao, X. 2016a. Diagnosis of complex active systems with uncertain temporal observations. In Buccafurri, F.; Holzinger, A.; Tjoa, A. M.; and Weippl, E., eds., *Availability, Reliability, and Security in Information Systems*, volume 9817 of *Lecture Notes in Computer Science*. Springer International Publishing AG Switzerland. 45–62.

Lamperti, G., and Zhao, X. 2016b. Viable diagnosis of complex active systems. In *IEEE International Conference on Systems, Man, and Cybernetics (SMC 2016)*, 457–462.

Lamperti, G.; Zanella, M.; and Zhao, X. 2018. *Introduction to Diagnosis of Active Systems*. Springer International Publishing.

Licata, I., and Sakaji, A. 2008. *Physics of Emergence and Organization*. World Scientific.

Mozetič, I. 1992. Hierarchical diagnosis. In Hamscher, W.; Console, L.; and de Kleer, J., eds., *Readings in Model-Based Diagnosis*. Morgan Kaufmann.

Paoli, A., and Lafortune, S. 2008. Diagnosability analysis of a class of hierarchical state machines. *Journal of Discrete Event Dynamic Systems: Theory and Applications* 18(3):385–413.

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.

Schumann, A., and Huang, J. 2008. A scalable jointree algorithm for diagnosability. In *Twenty-Third National Conference on Artificial Intelligence (AAAI 2008)*, 535–540.

Shenoy, P., and Shafer, G. 1986. Propagating belief functions with local computations. *IEEE Expert* 1(3):43–52.

Siddiqi, S., and Huang, J. 2007. Hierarchical diagnosis of multiple faults. In *20th International Joint Conference on Artificial Intelligence (IJCAI'07)*, 581–586.

Stumptner, M., and Wotawa, F. 2001. Diagnosing tree-structured systems. *Artificial Intelligence* 127(1):1–29.

Yoo, T., and Lafortune, S. 2002. Polynomial-time verification of diagnosability of partially observed discrete-event systems. *IEEE Transactions on Automatic Control* 47(9):1491–1495.