

# Predictability of Event Occurrences in Partially-Observed Discrete-Event Systems <sup>1</sup>

Sahika Genc <sup>2</sup> Stéphane Lafortune <sup>3</sup>

---

## Abstract

This paper studies the problem of predicting occurrences of a significant event in a partially-observed discrete-event system. The predictability of occurrences of an event in a system is defined in the context of formal languages. The predictability of a language is a stronger condition than the diagnosability of the language. Two necessary and sufficient conditions for predictability of occurrences of an event in systems modeled by regular languages are presented. Both conditions can be algorithmically tested. The first condition employs diagnosers. The second condition employs verifiers and results in a polynomial-time (in the number of states) complexity test for verification of predictability. When predictability holds, diagnosers can be used online to predict the significant event.

*Key words:* Discrete-event systems, prediction, diagnosis.

---

## 1 Introduction

This paper addresses the problem of predicting occurrences of a significant (e.g., fault) event in a discrete-event system (DES). The system under consideration is modeled by a language over an event set. The event set is partitioned into observable events (e.g., sensor readings, changes in sensor readings) and unobservable events, i.e., the events that are not directly recorded by the sensors attached to the system. The objective is to predict occurrences of a possibly unobservable event in the system behavior, based on the strings of observable events. If it is possible to predict occurrences of a given event in the system, then the role of the system operator can be more proactive than passive. Depending on the nature of the event, the system operator can be warned and the operator may decide to halt the system or otherwise take preventive measures. The two main contributions of this paper are: (i) the formal definition of predictability in the context of partially-observed DES and (ii) the devel-

opment of two separate tests to verify predictability of event occurrences. One of the tests provides a comprehensive setting that may help the operator track the system evolution online to further investigate the behavior of the system. The other test is computationally more efficient in the context of offline analysis.

The problem of prediction studied in this paper is inspired by the problem of event diagnosis in DES. The problem of event diagnosis in DES has received considerable attention in the last decade and diagnosis methodologies based on the use of discrete-event models have been successfully used in a variety of technological systems ranging from document processing systems to intelligent transportation systems. A discrete-event process called *diagnoser* introduced in [20] is of particular relevance to the present work. Specifically, the diagnoser is used to derive a necessary and sufficient condition for predictability in systems modeled by *regular* languages, as well as to perform prediction online.

To the best of our knowledge, the notion of predictability that is introduced and studied in this paper is different from prior works on other notions of predictability in [2,1,21,6]. For instance, the prediction problem considered in [2] is related to the properties of a special type of projection between two languages (sets of trajectories); this is much more general than our objective, which is to predict occurrences of specific events, but our work is not a special case of that in [2]. The state prediction of coupled automata studied in [1] is formulated as computing the state vector of  $n$  identical automata after  $T$

---

<sup>1</sup> This research was supported in part by NSF grant CCR-0325571, ECCS-0624821 and by ONR grant N00014-03-1-0232. The first author wishes to acknowledge support from a Barbour Fellowship from the Horace H. Rackham School of Graduate Studies at the University of Michigan.

<sup>2</sup> Sahika Genc is with General Electric Global Research Center, 1 Research Circle, Niskayuna, NY 12309 [gencs@ge.com](mailto:gencs@ge.com)

<sup>3</sup> Stéphane Lafortune is with the University of Michigan, 1301 Beal Avenue, Ann Arbor, MI 48109-2122 [stephane@eecs.umich.edu](mailto:stephane@eecs.umich.edu)

steps in the operation of the system; the system structure in this work is different from ours. The notion of prediction considered in [21] differs from the one in our work in the sense that in [21] predictability of a system is a necessary condition for diagnosability of the system while in our work diagnosability is a necessary condition for predictability. The prediction problem studied in [6] considers issuing a warning when it is *likely* for a fault to happen in the future evolution of the system; in our work, if the occurrence of an event is predictable in a language, then it is *certain* that the event will occur. Also, in [6], it is possible that false *fault prediction warnings* are issued; in our work, no false positives are issued. In [23], the authors develop a diagnostic system architecture that integrates the modeling, prediction and diagnostics components for a hybrid system with interacting continuous and discrete dynamics using stochastic Petri nets. In our case, the system model is logical and contains only discrete dynamics. Other papers concerned with predictability in DES or discrete-event simulation systems include [4,3,5,16,12,7].

This paper is organized as follows. In Section 2, the notation and frequently used terms are introduced. In Section 3, the predictability of occurrences of an event in a system is defined in the context of formal languages. The predictability property of a language is a stronger condition than the diagnosability of the language as defined in [20]. In Section 4, it is shown that in the case of regular languages, there exist necessary and sufficient conditions for predicting occurrences of an event in the language in the form of tests on diagnosers and verifiers. In Section 5, examples of applications of the theory of predictability for analysis of a Heating, Ventilation, and Air-Conditioning (HVAC) system and computer intrusion prediction are presented. In Section 6, a summary of the results in the paper is presented and concluding remarks are given. A preliminary and partial version of the results in this paper, excluding the polynomial-time test based on verifiers, was presented in [11].

## 2 Preliminaries

Let  $\Sigma$  be a finite set of events. A *string* is a finite-length sequence of events in  $\Sigma$ . Given a string  $s$ , the length of  $s$  (number of events including repetitions) is denoted by  $\|s\|$ . The set of all strings formed by events in  $\Sigma$  is denoted by  $\Sigma^*$ . The set  $\Sigma^*$  is also called the Kleene-closure of  $\Sigma$ . Any subset of  $\Sigma^*$  is called a *language* over  $\Sigma$ . Let  $L$  be a language over  $\Sigma$ . The *prefix-closure* of language  $L$  is denoted by  $\bar{L}$ . Given a string  $s \in L$ ,  $L/s$  is called the *post-language* of  $L$  after  $s$  and defined as  $L/s = \{t \in \Sigma^* : st \in L\}$ .  $L$  is live if every string in  $L$  can be extended to another string in  $L$ .  $\Sigma$  is partitioned as  $\Sigma = \Sigma_o \dot{\cup} \Sigma_{uo}$ , where  $\Sigma_o$  and  $\Sigma_{uo}$  denote the observable and unobservable events, respectively.

The projection of strings from  $L$  to  $\Sigma_o^*$  is denoted by

$P$ . Given a string  $s \in L$ ,  $P(s)$  is obtained by removing unobservable events (elements of  $\Sigma_{uo}$ ) in  $s$ . The inverse projection of a string  $s_o \in \Sigma_o^*$ , denoted by  $P^{-1}(s_o)$ , is the set of strings in  $\Sigma^*$  whose projection is equal to  $s_o$ . Formally,  $P^{-1}(s_o) = \{s \in \Sigma^* : P(s) = s_o\}$ . Given an event  $\sigma \in \Sigma$  and a string  $s \in \Sigma^*$ , we use the set notation  $\sigma \in s$  to denote that  $\sigma$  appears at least once in  $s$ . Let  $L$  be a prefix-closed and live language over  $\Sigma$ . Given an event  $\sigma \in \Sigma$  and  $L$ ,  $\Psi(\sigma, L)$  is the set of strings in  $L$  that end with  $\sigma$ . Formally,  $\Psi(\sigma, L) = \{s\sigma \in L : s \in \Sigma^*, \sigma \in \Sigma\}$ .

## 3 Definition of Predictability

In this section, we define the problem of predicting occurrences of an event in a partially-observed DES. We model the system as a language  $L$  over an event set  $\Sigma$ , with observable event set  $\Sigma_o$ . The event to be predicted may be an unobservable event or an observable one. The objective is to predict the occurrence of this event *before* it actually happens, based on the system model and the record of observable events. First, we present an illustrative example to introduce the notion of predictability. Then, we give the formal definition of the predictability of the occurrence of an event. We conclude the section by comparing the diagnosability of a language  $L$  as defined in [20] to the predictability of  $L$ .

Roughly speaking, the occurrence of an event in a language is predictable if it is possible to infer about future occurrences of the event based on the observable record of strings that do not contain the event to be predicted. Consider any string  $s$  in  $\Psi(\sigma_p, L)$  where  $\sigma_p$  is the event to be predicted. We wish to find a prefix  $t$  of  $s$  such that  $t$  does not contain  $\sigma_p$  and all the *long enough* continuations in  $L$  of the strings with the same projection as  $t$  contain  $\sigma_p$ . If there is at least one such  $t$  for each  $s$  containing  $\sigma_p$ , then every occurrence of  $\sigma_p$  becomes predictable in  $L$  once the corresponding  $t$  has been executed. Note that the definition of predictability should not dictate the occurrences of the event to be predicted in all strings in the language.

Consider the prefix-closed language  $L_1 = \overline{acfb^* + eab^*}$  generated by the FSA (Finite State Automaton)  $G_1$  shown in Fig. 1, where  $\Sigma_{uo} = \{e, f\}$  and  $\Sigma_o = \{a, b, c\}$ . Let  $f$  be the event to be predicted;  $f$  could be a fault event or some other significant event. If we observe  $a$ , we are not sure if  $f$  is going to happen since both  $ea$  and  $a$  result in the observation of  $a$  but only  $a$  has continuations containing  $f$ . If we observe  $ac$ , then we know for sure that  $f$  will occur before it is executed by the system. Thus,  $f$  is predictable in  $L_1$ . Consider the prefix-closed language  $L_2 = \overline{(a+b)fcb^* + eab^*}$  generated by the FSA  $G_2$  shown in Fig. 2, where  $\Sigma_{uo} = \{e, f\}$  and  $\Sigma_o = \{a, b, c\}$ . Let  $f$  be the event to be predicted. If  $b$  is observed, then we know for sure that  $f$  will occur. However, if  $a$  occurs, then we are not sure that  $f$  will occur and if  $ac$  occurs we know that  $f$  has already happened.

Thus,  $f$  is not predictable in  $L_2$  even if we know that  $f$  will occur for sure after the observation of  $b$ .

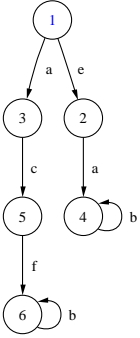


Fig. 1. FSA  $G_1$

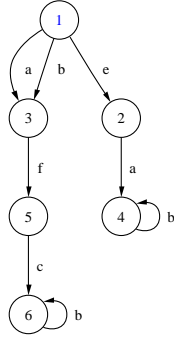


Fig. 2. FSA  $G_2$

We formally define the notion of predictability. The liveness assumption in the definition can be relaxed by adding self-loops at terminal states, as is done in [18] for instance. We omit this technical modification to keep our presentation simpler.

**Definition 1** Given  $L$  a prefix-closed, live language over  $\Sigma$ , occurrences of event  $\sigma_p \in \Sigma$  are predictable in  $L$  with respect to  $P$  if

$$(\exists n \in \mathbb{N})(\forall s \in \Psi(\sigma_p, L))(\exists t \in \bar{s})[(\sigma_p \notin t) \wedge \mathbf{P}]$$

where  $\mathbf{P} : (\forall u \in L)(\forall v \in L/u)[(P(u) = P(t)) \wedge (\sigma_p \notin u) \wedge (\|v\| \geq n) \Rightarrow (\sigma_p \in v)]$ .

We now use Definition 1 to show that  $\sigma_p = f$  is not predictable in  $L_2$ . We must find an  $n \in \mathbb{N}$  and a  $t \in \bar{s}$  for all  $s \in \Psi(f, L_2)$  such that  $f \notin t$  and for all  $u$  and its continuations  $v \in L_2/u$  if

- $u$  records the same string of observable events as  $t$ , i.e.,  $P(t) = P(u)$ , and
- $u$  does not contain  $f$ , i.e.  $f \notin u$ , and
- $v$  is of length greater than or equal to  $n \in \mathbb{N}$ , i.e.  $\|v\| \geq n$ ,

then  $v$  contains  $f$ .

The set of strings that end with  $f$  is  $\Psi(f, L_2) = \{af, bf\}$ . Let  $s = af$ . Then,  $t \in \bar{a}$ . Let  $t = \epsilon$  and  $P^{-1}(t) \cap (\Sigma \setminus f)^* \cap L_2 = \{e\}$ . When  $u = e$ ,  $L_2/u = \overline{ab^*}$ . None of the strings in  $L_2/u$  contains  $f$ . Let  $t = a$  and  $P^{-1}(t) \cap (\Sigma \setminus f)^* \cap L_2 = \{ea, a\}$ . If  $u = ea$ , then  $L_2/u = \overline{b^*}$ . None of the strings in  $L_2/u$  contains  $f$ . If  $u = a$ , then  $L_2/u = \overline{fcb^*}$ . The strings in  $L_2/u$  that have length 1 and greater contain  $f$ . Thus, the condition  $\mathbf{P}$  is not satisfied for all  $u$  such that  $P(t) = P(u)$  and  $f \notin u$ . This means that there is a string  $s$  such that there is no  $t \in \bar{s}$  which satisfies the condition in the definition. Thus,  $f$  is not predictable in  $L_2$ .

Let us show that  $f$  is predictable in  $L_1$  using the definition of predictability. The set of strings that end with  $f$  is  $\Psi(f, L_1) = \{acf\}$ . Then,  $s = acf$  and  $t \in \overline{ac}$ . Start with  $t = \epsilon$ . Then,  $P^{-1}(t) \cap (\Sigma \setminus f)^* \cap L_1 = \{e\}$ . If  $u = e$ , then  $L_1/u = \overline{ab^*}$ . None of the strings in  $L_1/u$  contains  $f$ . Consider  $t = a$ . Then,  $P^{-1}(t) \cap (\Sigma \setminus f)^* \cap L_1 = \{ea, a\}$ . If  $u = ea$ , then  $L_1/u = \overline{b^*}$ . None of the string in  $L_1/u$  contains  $f$ . If  $u = a$ , then  $L_1/u = \overline{cfb^*}$ . The strings in  $L_1/u$  that have length 2 and greater contain  $f$ . Let us try  $t = ac$ . Then,  $P^{-1}(t) \cap (\Sigma \setminus f)^* \cap L_1 = \{ac\}$ . Then,  $u = ac$  and  $L_1/u = \{fb^*\}$ . All the strings of length 1 and greater in  $L_1/u$  contain  $f$ . Thus,  $t = ac$  works. So, for all  $s \in \Psi(f, L_1)$ , there exists a  $t$  that satisfies the conditions in Definition 1 and thus  $f$  is predictable in  $L_1$ .

### 3.1 Diagnosability vs. Predictability

The predictability of occurrences of event  $\sigma_p$  in a language  $L$  is stronger than the diagnosability of  $L$  with respect to  $\sigma_p$ . We consider the diagnosability as defined in [20] in the context of formal languages. Roughly speaking,  $L$  is diagnosable with respect to  $\sigma_p$  if it is possible to detect occurrences of  $\sigma_p$  with a finite delay. For the sake of completeness, we recall in Definition 2 the formal definition of diagnosability.

**Definition 2** A prefix-closed and live language is diagnosable with respect to  $P$  and  $\sigma_p$  if

$$(\exists n \in \mathbb{N})(\forall s \in \Psi(\sigma_p, L))(\forall t \in L/s)[\|t\| \geq n \Rightarrow \mathbf{D}]$$

where  $\mathbf{D} : \omega \in P^{-1}P(st) \cap L \Rightarrow \sigma_p \in \omega$ .

An example of a language that is diagnosable with respect to an event but where the occurrence of the event is not predictable is language  $L_2$  generated by  $G_2$  in Fig. 2. Earlier in this section, we showed that event  $f$  is not predictable in  $L_2$ . However, event  $f$  is diagnosable in  $L_2$  with respect to Definition 2. After the observation of  $bc b$ , we know for sure that  $f$  has occurred once. The following remark follows directly from the above definitions.

**Proposition 3** Given a prefix-closed and live language  $L \subseteq \Sigma^*$ , if occurrences of  $\sigma_p \in \Sigma$  are predictable in  $L$  with respect to  $P$ , then  $L$  is diagnosable with respect to  $P$  and  $\sigma_p$ .

**PROOF.** Pick  $s_1 \in \Psi(\sigma_p, L)$ . By Definition 1, there exists  $n_1 \in \mathbb{N}$  and  $z_1 \in \bar{s}_1$  such that  $\sigma_p \notin z_1$  and  $\mathbf{P}$  is satisfied. We need to show that for all  $t_1 \in L/s_1$  if  $\|t_1\| \geq n$  for some positive integer  $n$ , then for all  $\omega \in P^{-1}P(s_1 t_1) \cap L$ ,  $\omega$  contains  $\sigma_p$ . Let  $s_1 = z_1 z_2$ . If  $\omega \in P^{-1}P(s_1 t_1) \cap L$ , then  $\omega \in P^{-1}P(z_1)P^{-1}P(z_2 t_1) \cap L$ . Choose  $n$  such that for all  $\|t_1\| \geq n$ , if  $\omega \in P^{-1}P(s_1 t_1) \cap L$ ,  $\omega = \omega_1 \omega_2$ , and  $P(\omega_1) = P(z_1)$ , then  $\|\omega_2\| \geq n_1$ . Suppose that there exists  $\omega$  such that  $\sigma_p \notin \omega$ . Then,

$\sigma_p \notin \omega_1$  and  $\sigma_p \notin \omega_2$ . By condition **P** in Definition 1, for all  $v \in L/u$  if  $P(u) = P(z_1)$ ,  $\sigma_p \notin u$ , and  $\|v\| \geq n_1$ , then  $\sigma_p \in v$ . Thus,  $\sigma_p \in \omega_2$ . This is a contradiction. Thus, there is no  $\omega \in P^{-1}P(s_1t_1) \cap L$  such that  $\sigma_p \notin \omega$ . This completes the proof.  $\square$

#### 4 Verification of Predictability in Regular Languages

In this section, we consider systems modeled by regular languages. The notation we use for an FSA is a four-tuple

$$G = (Q, \Sigma, \delta, q_0) \quad (1)$$

where  $Q$  is the set of states,  $\Sigma$  is the finite set of events,  $\delta : Q \times \Sigma \rightarrow Q$  is the state transition function and  $q_0$  is the initial state. We present two necessary and sufficient conditions for predictability of occurrences of an event in systems modeled by regular languages. The first condition employs diagnosers. The second condition employs verifiers and results in a polynomial-time (in the number of states) complexity test for verification of predictability.

##### 4.1 Verification Using Diagnosers

The necessary and sufficient condition (presented later in this section) for predictability is based on a discrete-event process called *diagnoser*. The diagnoser is an FSA built for the system with respect to a projection  $P$  onto the set of observable events and to a given event. Let  $G = (Q, \Sigma, \delta, q_0)$  be an FSA that generates language  $L$ . We denote by  $D_G$  the diagnoser built for  $G$  and  $\sigma_p \in \Sigma$ . The diagnoser  $D_G$  is of the form

$$D_G = (Q_D, \Sigma_o, \delta_D, q_{D,0}, \sigma_p), \quad (2)$$

where  $Q_D$  is the set of diagnoser states,  $\delta_D : Q_D \times \Sigma_o \rightarrow Q_D$  is the diagnoser state transition function, and  $q_{D,0} \in Q_D$  is the initial diagnoser state. The diagnoser state space  $Q_D$  is a subset of  $2^{Q \times \{N, F1\}}$ . State  $q_D \in Q_D$  is of the form  $q_D = \{(q_1, l_1), \dots, (q_n, l_n)\}$ , where  $q_i \in Q$  and  $l_i \in \{N, F1\}$  for  $i = 1, \dots, n$ . The reason for using label  $F1$  instead of simply  $Y$  for instance is to match the notation of UMDES [15] and DESUMA [17], which are the software tools used for the examples in this paper. Label  $F1$  is to be interpreted as “event  $\sigma_p$  has occurred in reaching the current state,” while label  $N$  means that “event  $\sigma_p$  has not occurred in reaching the current state.” Also, we adopt the convention of not including the unobservable reach in a diagnoser state.<sup>4</sup>

<sup>4</sup> The literature is not consistent in this regard. The choice of including or not the unobservable reach may affect the structure of the diagnoser but has no implications on the results derived from it.

Let  $q_D$  and  $q'_D$  be two diagnoser states in  $Q_D$  such that  $q'_D$  is reached from  $q_D$  by  $\sigma_o \in \Sigma_o$ , i.e.,  $q'_D = \delta_D(q_D, \sigma_o)$  is defined. Let  $q_D = \{(q_1, l_1), \dots, (q_m, l_m)\}$  and  $q'_D = \{(q'_1, l'_1), \dots, (q'_n, l'_n)\}$ . For all  $i \in \{1, \dots, n\}$ , there exists  $j \in \{1, \dots, m\}$  such that  $q'_i = \delta(q_j, s)$ , where  $s = t\sigma_o$  and  $t \in \Sigma_{u_o}^*$ , and

$$l'_i = \begin{cases} F1, & \text{if } l_j = F1 \text{ or } (\sigma_p \in s), \\ N, & \text{if } l_j = N \text{ and } (\sigma_p \notin s). \end{cases} \quad (3)$$

We say that a diagnoser state  $q_D = \{(q_1, l_1), \dots, (q_m, l_m)\} \in Q_D$  for  $m \in \mathbb{N}$  is: *normal* if  $l_j = N$  for all  $j = 1, \dots, m$ ; *certain* if  $l_j = F1$  for all  $j = 1, \dots, m$ ; and *uncertain* if there exist  $l_j = N$  and  $l_i = F1$  for some  $i, j \in \{1, \dots, m\}$ . We denote by  $Q_D^N \subseteq Q_D$  the set of diagnoser states that are normal, by  $Q_D^U \subseteq Q_D$  the set of diagnoser states that are uncertain, and by  $Q_D^C \subseteq Q_D$  the set of diagnoser states that are certain.

The diagnosers<sup>5</sup> built for the preceding  $G_1$  and  $G_2$  are shown in Fig. 3 and Fig. 4, respectively, where  $\sigma_p = f$ . The diagnoser state  $\{(6, F1)\}$  is a certain state while  $\{(3, N), (4, N)\}$  is a normal state.

In the previous section, we proved that  $f$  is predictable in  $L_1$  but not in  $L_2$ , where  $L_1$  and  $L_2$  are the languages generated by  $G_1$  and  $G_2$ , respectively. Consider the diagnoser  $D_{G_1}$  in Fig. 3. Upon observation of  $ac$ , we transition to the diagnoser state  $\{(5, N)\}$  and all the diagnoser states accessible from  $\{(5, N)\}$ , namely,  $\{(6, F1)\}$ , are in a cycle of certain states. Thus, upon observation of  $ac$ , we know *for sure* that  $f$  will occur in the future. However, if we consider the diagnoser  $D_{G_2}$  in Fig. 4, there is no such state. There are two normal diagnoser states  $\{(3, N)\}$  and  $\{(3, N), (4, N)\}$  that lead to the certain state  $\{(6, F1)\}$ ; once in that state,  $D_{G_2}$  stays there. However, there is a transition from  $\{(3, N), (4, N)\}$  that leads to a normal state and subsequent transitions stay in that normal state. This means that upon observation of  $a$ , we are not sure if  $f$  will happen or not in the future. From these two examples, we conclude that the normal states leading to specific cycles play a role in the predictability of the occurrences of an event. In the following, we present the notions that will lead us to state the necessary and sufficient condition for predictability.

We define an accessibility operation on an FSA to find the accessible part of the FSA from a state.

**Definition 4** Let  $G = (Q, \Sigma, \delta, q_0)$  and  $q \in Q$ . The accessible part of  $G$  with respect to  $q$  is denoted by  $Ac(G, q)$

<sup>5</sup> The diagnosers shown in this paper are built using DESUMA [17]. The notation used for diagnoser states in the program is a compressed form, e.g., “3N” instead of “ $\{(3, N)\}$ .”

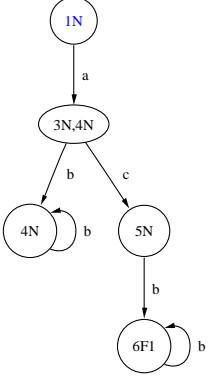


Fig. 3. Diagnoser of  $G_1$

and is

$$Ac(G, q) = (Q_{ac}, \Sigma, \delta_{ac}, q), \quad (4)$$

where  $Q_{ac} = \{q' \in Q : (\exists s \in \Sigma^*)(\delta(q, s) = q' \text{ is defined})\}$ , and  $\delta_{ac} = \delta|_{Q_{ac} \times \Sigma \rightarrow Q_{ac}}$ .

Let  $G = (Q, \Sigma, \delta, q_0)$ . We say that a set of states  $\{q_1, q_2, \dots, q_n\} \subseteq Q$  and a string  $\sigma_1 \sigma_2 \dots \sigma_n \in \Sigma^*$  form a *cycle* if  $q_{i+1} = \delta(q_i, \sigma_i)$ ,  $i = 1, \dots, n-1$  and  $q_1 = \delta(q_n, \sigma_n)$ .

In the rest of this section, we assume the system satisfies the following: *If  $\{q_1, q_2, \dots, q_n\} \subseteq Q$  and  $\sigma_1 \sigma_2 \dots \sigma_n \in \Sigma^*$  form a cycle, then there exists at least one observable event  $\sigma_j$  in  $\{\sigma_1, \dots, \sigma_n\} \subseteq \Sigma$ .* That is,  $G$  does not contain a cycle in which states are connected with unobservable events only.

Lemma 5 below states that if there is a cycle in  $D_G$  that contains a certain diagnoser state, then all the diagnoser states in the cycle are certain (since the  $F1$  label propagates). Lemma 6 states that if there is a cycle in  $D_G$  that is formed by uncertain or normal states, then there exists a corresponding cycle in  $G$  such that all the states in the cycle have normal labels in the cycle in  $D_G$ . The proofs of Lemmas 5 and 6 are straightforward from the results in [19] and omitted.

**Lemma 5** *Let  $G = (Q, \Sigma, \delta, q_0)$  be an FSA that generates  $L$  such that  $L$  is prefix-closed and live, let  $D_G = (Q_D, \Sigma_o, \delta_D, q_{D,0}, \sigma_p)$  be the diagnoser for  $G$  and  $\sigma_p$ . Suppose  $\{q_{D,1}, \dots, q_{D,n}\} \subseteq Q_D$  and  $\sigma_{o,1} \dots \sigma_{o,n} \in \Sigma_o^*$  form a cycle in  $D_G$  where  $n \in \mathbb{N}$ . If there exists  $i \in \{1, \dots, n\}$  such that  $q_{D,i} \in Q_D^C$ , then  $q_{D,j} \in Q_D^C$  for all  $j = 1, \dots, n$ .*

**Lemma 6** *Let  $G = (Q, \Sigma, \delta, q_0)$  be an FSA that generates  $L$  such that  $L$  is prefix-closed and live, and let  $D_G = (Q_D, \Sigma_o, \delta_D, q_{D,0}, \sigma_p)$  be the diagnoser for  $G$  and  $\sigma_p$ . Suppose  $\{q_{D,1}, \dots, q_{D,n}\} \subseteq Q_D$  and  $\sigma_{o,1} \dots \sigma_{o,n} \in \Sigma_o^*$  form a cycle in  $D_G$  where  $n \in \mathbb{N}$  and  $q_{D,i}$  is in  $Q_D^U$  or  $Q_D^N$  for all  $i = 1, 2, \dots, n$ . Then, there exists  $(q_i, l_i) \in q_{D,i}$  for  $i = 1, \dots, n$ , such that  $q_{i+1} = \delta(q_i, s_i)$  for  $i = 1, \dots, n-1$  and  $q_1 = \delta(q_n, s_n)$  where  $s_i \in \Sigma^*$ ,  $P(s_i) = \sigma_{o,i}$ , and  $l_i = N$  for  $i = 1, \dots, n$ .*

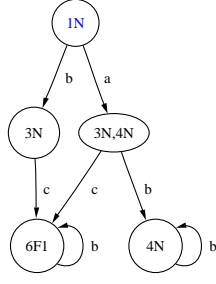


Fig. 4. Diagnoser of  $G_2$

Let  $F_D$  be the set of normal diagnoser states that possess an immediate successor that is not normal. For example,  $F_D = \{(5, N)\}$  in  $D_{G_1}$  and  $F_D = \{(3, N), \{(3, N), (4, N)\}\}$  in  $D_{G_2}$ . Formally,

$$F_D = \{x_D \in Q_D^N : \exists y_D = \delta_D(x_D, \sigma_o) \text{ such that } \sigma_o \in \Sigma_o \text{ and } y_D \notin Q_D^N\}. \quad (5)$$

Lemma 7 states that any uncertain or certain diagnoser state is reached from a diagnoser state in  $F_D$ . The proof is by induction on the sequence of observable events; it is relatively straightforward and has been omitted.

**Lemma 7** *Let  $G = (Q, \Sigma, \delta, q_0)$  be an FSA that generates  $L$  such that  $L$  is prefix-closed and live, and let  $D_G = (Q_D, \Sigma_o, \delta_D, q_{D,0}, \sigma_f)$  be the diagnoser for  $G$  and  $\sigma_p$ . Let  $x_{D,i} = \delta_D(x_{D,i-1}, \sigma_{o,i})$  for  $i = 1, \dots, m$  where  $m \in \mathbb{N}$ ,  $x_{D,i}$  is a diagnoser state,  $\sigma_{o,i}$  is an observable event for  $i = 1, \dots, m$ , and  $x_{D,0}$  is the initial diagnoser state. If  $x_{D,m}$  is in  $Q_D^U$  or  $Q_D^C$ , then there exists  $M \leq m$  such that  $x_{D,M} \in F_D$ .*

In the following theorem, we state the necessary and sufficient condition for predictability of occurrences of an event. The condition is based on analyzing the cycles in the diagnoser.

**Theorem 8** *Let  $G = (Q, \Sigma, \delta, q_0)$  be an FSA that generates  $L$  where  $L$  is prefix-closed and live. Let  $D_G = (Q_D, \Sigma_o, \delta_D, q_{D,0}, \sigma_p)$  be the diagnoser for  $G$  and  $\sigma_p$ . The occurrences of  $\sigma_p$  are predictable in  $L$  with respect to  $P$  iff for all  $q_D \in F_D$ , condition **C** holds, where*

**C** : *all cycles in  $Ac(D_G, q_D)$  are cycles of certain diagnoser states.*

**PROOF.** The proof is in two parts.

( $\Rightarrow$ ): We prove that if  $\sigma_p$  is predictable in  $L$ , then for all  $q_D \in F_D$  the only cycles in  $Ac(D_G, q_D)$  are cycles of certain diagnoser states. The proof is by contradiction. Suppose that there exists  $q_D \in F_D$  such that  $Ac(D_G, q_D)$  contains a cycle formed by  $\{x_{D,1}, \dots, x_{D,m}\}$  and  $\sigma_{o,1} \dots \sigma_{o,m} \in \Sigma_o^*$  where  $x_{D,i} \notin Q_D^C$  for some  $i \in \{1, 2, \dots, m\}$ .

By Lemma 5, if there exists a diagnoser state  $x_{D,i}$  in the cycle such that  $x_{D,i}$  is not a certain diagnoser state, then none of the other diagnoser states in the cycle are certain. Thus,  $x_{D,i} \notin Q_D^C$  for all  $i = 1, 2, \dots, m$ . By Lemma 6, corresponding to the cycle of diagnoser states in the diagnoser, there exists a cycle in  $G$  such that each state in that cycle is labeled with  $N$  in the cycle in the diagnoser. Suppose that the cycle in  $G$  is formed by  $\{x_1, \dots, x_m\}$  and  $s_1 \dots s_m \in \Sigma^*$  where  $(x_i, N) \in x_{D,i}$  and  $\omega_i \in \Sigma^*$  such that  $P(\omega_i) = \sigma_{o,i}$  for  $i = 1, 2, \dots, m$ .

Let  $q_D \in F_D$  be reached from the initial diagnoser state  $q_{D,0}$  by  $s_o \in \Sigma_o^*$ . Since  $q_D$  is in  $F_D$ , then there exists  $s \in \Psi(\sigma_p, L)$  such that  $P(s) = s_o$ . We wish to show that for all  $t \in \bar{s}$ ,  $(\sigma_p \notin t) \wedge \mathbf{P}$  is satisfied. In order to prove that  $\mathbf{P}$  is violated, we wish to find a  $u \in L$  and  $v \in L/u$  such that  $P(u) = P(t)$  and  $\sigma_p \notin u$ , and if  $v$  is of length greater than any  $n \in \mathbb{N}$ , then  $v$  does not contain  $\sigma_p$ . It is sufficient to prove the theorem by considering a particular  $t \in \bar{s}$ . Let  $s = s_1\sigma_p$  where  $s_1 \in \Sigma^*$ . If the condition,  $\mathbf{P}$ , is violated for  $t_1 = s_1$ , then it is violated for all  $t \in \bar{t}_1$ . This is because if there is a long enough suffix of  $t_1$  violating the condition,  $\mathbf{P}$ , then that suffix can be used to prove that there is a long enough suffix of any  $t \in \bar{t}$  violating  $\mathbf{P}$ .

Pick a diagnoser state in the cycle. Without loss of generality pick  $x_{D,1}$ . Then, we pick the state in the diagnoser state which has label  $N$  and is a part of the corresponding cycle in  $G$ . Let  $(x_1, l_1)$  be that state in  $x_{D,1}$ , with  $l_1 = N$ .

Suppose that  $x_{D,1}$  is reached from  $q_D$  by executing  $s'_o \in \Sigma_o^*$ . Then,  $x_{D,1} = \delta_D(q_{D,0}, s_o s'_o)$ . Since  $x_1$  is in the corresponding cycle in  $G$ , then  $x_1 = \delta(x_1, (\omega_1 \dots \omega_m)^k)$  for  $k \in \mathbb{N}$  and  $k \geq n$ . Let  $u \in L$  and  $u' \in L/u$  such that  $x_1 = \delta(q_0, uu')$  and  $P(u) = s_o = P(t_1)$ . Then,  $x_1 = \delta(q_0, uu'(\omega_1 \dots \omega_m)^k)$ . Let  $v = u'(\omega_1 \dots \omega_m)^k$ . Since  $x_1$  has normal label, then neither  $u$  nor  $u'$  does not contain  $\sigma_p$ . Also, by Lemma 6, for  $i = 1, \dots, m$ ,  $\omega_i \in \Sigma^*$  does not contain  $\sigma_p$ . Thus,  $v$  does not contain  $\sigma_p$ . This violates the condition  $\mathbf{P}$  in the definition of predictability. Thus, there is a contradiction. This completes one part of the proof.

( $\Leftarrow$ ): We prove that if for all  $q_D \in F_D$  the only cycles in  $Ac(D_G, q_D)$  are cycles of certain diagnoser states, then  $\sigma_p$  is predictable in  $L$ .

Pick any  $s \in \Psi(\sigma_p, L)$ . Let  $q = \delta(q_0, s) \in Q$ . Then, pick any  $s_{uo}\sigma_o \in L/s$  such that  $s_{uo} \in \Sigma_{uo}^*$  and  $\sigma_o \in \Sigma_o$ . Let  $y = \delta(q, s_{uo}\sigma_o) \in Q$ . Suppose that  $P(s) = s_o \in \Sigma_o^*$ . Then, let  $x_D = \delta_D(q_{D,0}, s_o)$  and  $y_D = \delta_D(x_D, \sigma_o)$  in  $Q_D$ . Then, there exists  $(y, l_y) \in y_D$  where  $l_y = F1$ . Thus,  $y_D \in Q_D^U \cup Q_D^C$ . We now consider the following two cases: (i)  $x_D \in Q_D^N$ , thus,  $x_D \in F_D$ , and (ii)  $x_D \in Q_D^U \cup Q_D^C$ .

*Case (i).* Since  $x_D \in Q_D^N$  and  $y_D \notin Q_D^N$ , then  $x_D \in F_D$ . We choose  $t = s$ . For all  $u$  such that  $P(u) = P(t)$ ,  $P(u) = s_o$ . Since the only cycles in  $Ac(D_G, x_D)$  are cycles of certain states, then for all  $v \in L/u$ ,  $v$  contains  $\sigma_p$ .

*Case (ii).* If  $x_D \in Q_D^U \cup Q_D^C$ , i.e.,  $x_D$  is not normal, then we wish to find a normal diagnoser state in  $F_D$  from which  $x_D$  is reached. By Lemma 7, there exists a diagnoser state  $w_D$  reachable from the initial diagnoser state,  $x_D$  is accessible from  $w_D$ , and  $w_D$  is in  $F_D$ . Then, since  $F_D$  consists of normal diagnoser states,  $w_D$  is in  $Q_D^N$ . Thus, the proof of Case (ii) reduces to the case of

(i) in which we substitute  $w_D \in Q_D^N$  for  $x_D \in Q_D^N$ . This completes the second part of the proof.  $\square$

Let us use Theorem 8 to assess the predictability of  $f$  in  $L_1$  and  $L_2$ . (In the previous section, we used Definition 1 to prove that  $f$  is predictable in  $L_1$  but not in  $L_2$ .) First, we determine that  $F_D = \{(5, N)\}$  in  $D_{G_1}$  and  $F_D = \{(3, N), \{(3, N), (4, N)\}\}$  in  $D_{G_2}$ . Then  $Ac(D_{G_1}, (5, N))$  contains a single cycle that is a cycle of certain states. Thus, based on the theorem,  $f$  is predictable in  $L_1$ . However,  $Ac(D_{G_1}, \{(3, N), (4, N)\})$  contains a cycle of normal states, i.e.,  $\{(4, N)\}$ . So,  $f$  is not predictable in  $L_2$ .

### Remarks:

*1 - Improvement of the Test:* It is sufficient to test condition  $\mathbf{C}$  in Theorem 8 on certain subsets of  $F_D$  to guarantee that this condition holds for all states in  $F_D$ . The sufficient condition follows from reachability arguments. If there are two states in  $F_D$  such that one is reachable from the other, then testing the condition on the originating state is sufficient. This leads to a potential improvement because the set of these originating states may be smaller in size compared to  $F_D$ . An algorithm exploiting this property can be derived; the details are omitted here but can be found in [8].

*2 - Online Prediction:* Online implementation of event prediction is done using diagnosers. For predictable systems, once a state in  $F_D$  is reached, the diagnoser can declare that event  $\sigma_p$  will occur at some point in the future.

### 4.2 Verification Using Verifiers

In this section, we define another discrete-event process called *verifier*. We present a necessary and sufficient condition for predictability based on the verifier. The use of verifiers for offline testing of predictability is computationally efficient. The computational complexity of the test based on verifiers is polynomial-time. On the other hand, the complexity of the test based on diagnosers is exponential-time in the worst case.

Verifiers were first defined in [22]. In [22], the authors use verifiers to test for diagnosability. The verifier is a nondeterministic FSA built for the system with respect to a projection  $P$  onto the set of observable events,  $\Sigma_o$ , and a set of significant events (in our case, the event to be predicted,  $\sigma_p$ ). Let  $G = (Q, \Sigma, \delta, q_0)$  be an FSA that generates language  $L$ . We denote by  $V_G$  the verifier built for  $G$  and  $\sigma_p$ . The verifier  $V_G$  is of the form

$$V_G = (Q_V, \Sigma, \delta_V, q_{V,0}, \sigma_p), \quad (6)$$

where  $Q_V$  is the set of verifier states,  $\delta_V$  is the verifier state transition function, and  $q_{V,0}$  is the initial verifier

state. Verifier state  $q_V \in Q_V$  is of the form

$$q_V = [(q_1, l_1), (q_2, l_2)], \quad (7)$$

where  $q_i \in Q$  and  $l_i \in \{N, F1\}$  for  $i = 1, 2$ . The verifier state space  $Q_V$  is a subset of  $Q \times \{N, F1\} \times Q \times \{N, F1\}$ .

Let  $q_V = [(q_1, l_1), (q_2, l_2)] \in Q_V$ . The state transition function  $\delta_V(q_V, \sigma)$  is defined for some  $\sigma \in \Sigma$  if  $\delta(q_1, \sigma)$  or  $\delta(q_2, \sigma)$  is defined. Suppose that  $\delta_V(q_V, \sigma)$  is defined for some  $\sigma \in \Sigma$ . Since  $V_G$  is nondeterministic, then  $\delta_V(q_V, \sigma)$  is a set of verifier states, and is defined as follows:

- If  $\sigma \in \Sigma_{u_o}$ , then

$$\delta_V([(q_1, l_1), (q_2, l_2)], \sigma) = \{[(\delta(q_1, \sigma), l'_1), (q_2, l_2)], [(q_1, l_1), (\delta(q_2, \sigma), l'_2)], [(\delta(q_1, \sigma), l'_1), (\delta(q_2, \sigma), l'_2)]\} \quad (8)$$

- If  $\sigma \in \Sigma_o$ , then

$$\delta_V([(q_1, l_1), (q_2, l_2)], \sigma) = [(\delta(q_1, \sigma), l'_1), (\delta(q_2, \sigma), l'_2)], \quad (9)$$

where if  $\sigma = \sigma_p$ , then  $l'_1 = l'_2 = F1$ , otherwise  $l'_1 = l_1$  and  $l'_2 = l_2$ .

The verifiers built for  $G_1$  and  $G_2$  (in Fig. 1 and Fig. 2) where  $\sigma_p = \{f\}$  are shown in Fig. 5 and Fig. 6. The verifiers shown in this paper are built using DESUMA. The *verifier algorithm* of the DESUMA software library builds an equivalent minimal verifier. As a result some of the branches described in the formal definition of the verifier transition function do not appear in the figures. Also, note that the verifier may contain deadlocks if the observable event is not feasible from both of the state components in the verifier state, e.g.,  $[(4, N), (3, N)]$  in  $V_{G_1}$  and  $[(4, N), (5, F1)]$  in  $V_{G_2}$ .

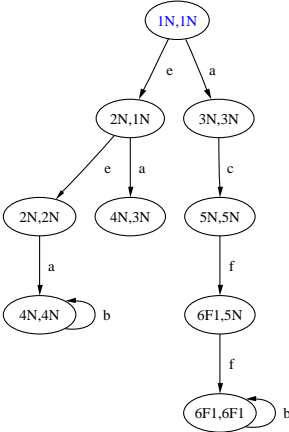


Fig. 5. Verifier of  $G_1$

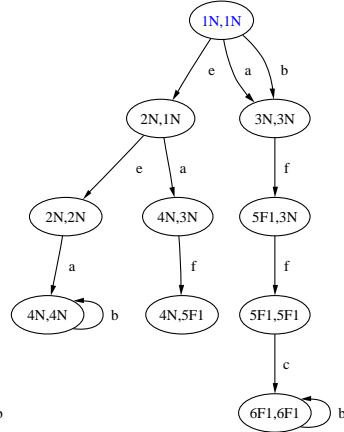


Fig. 6. Verifier of  $G_2$

**Lemma 9** Let  $G = (Q, \Sigma, \delta, q_0)$  be an FSA that generates  $L$ , let  $s = u_s \sigma_o$  and  $t = u_t \sigma_o$  in  $L$  be such that

$q_s = \delta(q_0, s)$  and  $q_t = \delta(q_0, t)$  where  $u_s, u_t \in \Sigma_{u_o}^*$  and  $\sigma_o \in \Sigma_o$ , and let  $V_G = (Q_V, \Sigma, \delta_V, q_{V,0}, \sigma_p)$  be the verifier for  $G$  and  $\sigma_p$ . Then, there exists  $q_V \in Q_V$  such that  $q_V \in \delta_V(q_{V,0}, u_s u_t \sigma_o)$  is defined and  $q_V = [(q_s, l_s), (q_t, l_t)]$ , where  $l_s, l_t \in \{N, F1\}$ .

**PROOF.** By definition  $q_{V,0} = [(q_0, N), (q_0, N)]$ . Let  $q_{u_s} = \delta(q_0, u_s) \in Q$ . Since  $u_s$  is feasible from  $q_0$ , then it is also feasible from  $q_{V,0}$ . Thus, there exists  $q_{V,s} = [(q_{u_s}, l_{u_s}), (q_0, N)] \in Q_V$  where  $l_{u_s} \in \{N, F1\}$ . Let  $q_{u_t} = \delta(q_0, u_t) \in Q$ . Since  $u_s$  is feasible from  $q_0$ , then it is also feasible from  $q_{V,s}$ . Thus,  $q_{V,s} = [(q_{u_s}, l_{u_s}), (q_{u_t}, l_{t_s})] \in Q_V$  where  $l_{u_t} \in \{N, F1\}$ . The observable event  $\sigma_o$  is feasible from both  $q_{u_s}$  and  $q_{u_t}$ , then by definition of the verifier transition function, there exists  $q_V = [(q_s, l_s), (q_t, l_t)]$ , where  $l_s, l_t \in \{N, F1\}$ . This completes the proof.  $\square$

**Lemma 10** Let  $G = (Q, \Sigma, \delta, q_0)$  be an FSA that generates  $L$ , let  $s, t \in L$  such that  $q_s = \delta(q_0, s)$  and  $q_t = \delta(q_0, t)$  is defined, and let  $V_G = (Q_V, \Sigma, \delta_V, q_{V,0}, \sigma_p)$  be the verifier for  $G$  and  $\sigma_p$ . Then,

$$P(s) = P(t) \Leftrightarrow q_V = [(q_s, l_s), (q_t, l_t)] \in Q_V, \quad (10)$$

where  $l_s, l_t \in \{N, F1\}$ .

In Lemma 10, we say that if there are two strings in  $L$  that have the same sequence of observable events, then there exists a corresponding string in the verifier. Also, according to this result, there is a corresponding string in the verifier for every string in  $L$ , since each string has the same observation with itself. The proof of Lemma 10 follows from Lemma 9 and is an induction on the sequence of observable events. The proof is omitted.

We say that a verifier state  $q_V = [(q_1, l_1), (q_2, l_2)]$  is: *normal* if  $l_1 = l_2 = N$ ; *certain* if  $l_1 = l_2 = F1$ ; and *uncertain* if  $l_1 = F1$  and  $l_2 = N$  or vice versa. We denote by  $Q_V^N$  the set of verifier states that are normal, by  $Q_V^C$  the set of states that are certain, and by  $Q_V^U$  the set of states that are uncertain.

**Lemma 11** Let  $G = (Q, \Sigma, \delta, q_0)$  be an FSA that generates  $L$  such that  $L$  is prefix-closed and live, let  $V_G = (Q_V, \Sigma, \delta_V, q_{V,0}, \sigma_p)$  be the verifier for  $G$  and  $\sigma_p$ . Suppose  $\{q_{V,1}, \dots, q_{V,n}\} \subseteq Q_V$  and  $\sigma_1 \dots \sigma_n \in \Sigma^*$  form a cycle in  $V_G$  where  $n \in \mathbb{N}$ . If there exists  $i \in \{1, \dots, n\}$  such that  $q_{V,i} \in Q_V^C$ , then  $q_{V,j} \in Q_V^C$  for all  $j = 1, \dots, n$ .

Lemma 11 is a result of the fault label propagation in the verifier; its proof is omitted. We wish to form a set of verifier states  $F_V$  that will serve the same purpose as  $F_D$  does in the verification of predictability using diagnosers; in other words, we wish to draw the boundary of the change from normal verifier states to uncertain or certain verifier states. Due the structure of the verifier,

the formal definitions differ slightly from those in the diagnoser case.

A set of normal verifier states  $F'_V$  is defined as follows:

$$F'_V = \{x_V \in Q_V^N : \delta_V(x_V, s_{uo}\sigma_p) \text{ is defined for } s_{uo} \in \Sigma_{uo}^* \text{ and } \sigma_p \notin s_{uo}\}. \quad (11)$$

In words, the set  $F'_V$  contains the normal verifier states that reach a certain or uncertain verifier state through event  $\sigma_p$ , possibly preceded by unobservable events (distinct from  $\sigma_p$  when this event is itself unobservable). We then form the set  $F_V$  with the states in  $F'_V$  as follows:

$$F_V = \{q_V = [(x, l_x), (y, l_y)] \in Q_V^N : [(x, l_x), (., .)] \text{ or } [(., .), (y, l_y)] \in F'_V\} \quad (12)$$

where  $(., .)$  denotes any one of the state components in a verifier state. The need for the set  $F_V$  is to capture all verifier states that are reached by strings that have the same observable projection as strings that reach a state in  $F'_V$ . There could be verifier states in  $F_V$  that are not in  $F'_V$ ; such states correspond to strings that will not contain  $\sigma_p$  in any of their continuations, and hence will cause a violation of predictability. As an example,  $F'_V = F_V = \{[(5, N), (5, N)]\}$  in  $V_{G_1}$  and  $F'_V = \{[(3, N), (3, N)], [(4, N), (3, N)]\}$ ,  $F_V = \{[(3, N), (3, N)], [(4, N), (3, N)], [(4, N), (4, N)]\}$  in  $V_{G_2}$ .

**Lemma 12** *Let  $G = (Q, \Sigma, \delta, q_0)$  be an FSA that generates  $L$  such that  $L$  is prefix-closed and live, and let  $V_G = (Q_V, \Sigma_o, \delta_V, q_{V,0}, \sigma_p)$  be the verifier for  $G$  and  $\sigma_p$ . Let  $x_{V,i} = \delta_V(x_{V,i-1}, \sigma_{o,i})$  for  $i = 1, \dots, m$  where  $m \in \mathbb{N}$ ,  $x_{V,i}$  is a verifier state,  $\sigma_{o,i}$  is an observable event for  $i = 1, \dots, m$ , and  $x_{V,0}$  is the initial verifier state. If  $x_{V,m}$  is in  $Q_V^U$  or  $Q_V^C$ , then there exists  $M \leq m$  such that  $x_{V,M} \in F_V$ .*

The proof of Lemma 12 is similar to the proof of Lemma 7 and is omitted.

In the following theorem, we state our second necessary and sufficient condition for predictability of occurrences of an event. The condition is based on analyzing the cycles in the verifier.

**Theorem 13** *Let  $G = (Q, \Sigma, \delta, q_0)$  be an FSA that generates  $L$  where  $L$  is prefix-closed and live. Let  $V_G = (Q_V, \Sigma_o, \delta_V, q_{V,0}, \sigma_p)$  be the verifier for  $G$  and  $\sigma_p$ . The occurrences of  $\sigma_p$  are predictable in  $L$  with respect to  $P$  iff for all  $q_V \in F_V$ , condition  $\mathbf{C}_V$  holds, where*

$$\mathbf{C}_V : \text{all cycles in } Ac(V_G, q_V) \text{ are cycles of certain verifier states.}$$

**PROOF.** The proof is in two parts.

( $\Rightarrow$ ): We prove that if  $\sigma_p$  is predictable in  $L$ , then for all  $q_V \in F_V$  the only cycles in  $Ac(V_G, q_V)$  are cycles of certain verifier states. The proof is by contradiction.

Suppose that there exists  $q_V \in F_V$  such that  $Ac(V_G, q_V)$  contains a cycle formed by  $\{x_{V,1}, \dots, x_{V,m}\}$  and  $\sigma_1 \dots \sigma_m \in \Sigma^*$  where  $x_{V,i} \notin Q_V^C$  for some  $i \in \{1, \dots, m\}$ .

Let  $q_V = [(q_1, N), (q_2, N)] \in \delta_V(q_{V,0}, \omega_1)$  where  $q_1, q_2 \in Q$  and  $\omega_1 \in \Sigma^*$ . If  $q_V \in F_V$ , then there exist  $y_V$  and  $z_V$  such that  $y_V = \delta_V(q'_V, s_{uo})$  and  $z_V = \delta_V(y_V, \sigma_p)$  where  $s_{uo} \in \Sigma_{uo}^*$ ,  $\sigma_p \notin s_{uo}$  and  $q'_V = [(q_1, N), (., .)]$  or  $q'_V = [(., .), (q_2, N)]$ . The existence of  $y_V$  and  $z_V$  follows from Lemma 10 and the existence of  $q'_V$  follows from the definition of  $F_V$ .

There exists  $s \in \Psi(\sigma_p, L)$  such that  $P(s) = P(\omega_1 s_{uo} \sigma_p)$ . We wish to show that for all  $t \in \bar{s}$  such that  $\sigma_p \notin t$ , the condition,  $\mathbf{P}$ , is violated. Let  $s = s_1 \sigma_p$  where  $s_1 \in \Sigma^*$ . If the condition,  $\mathbf{P}$ , is violated for  $t_1 = s_1$ , then it is violated for all  $t \in \bar{t}_1$ . Thus, hereafter, we consider the case of  $t_1$  only. We pick without loss of generality  $x_{V,1}$  in the cycle. Let  $x_{V,1} = [(x_1, N), (x_2, l_{x_2})]$  where  $x_1, x_2 \in Q$  and  $l_{x_2} \in \{N, F1\}$ , and let  $x_{V,1} \in \delta_V(q_V, \omega_2)$ .

There exist  $u \in L$  and  $u' \in L/u$  such that  $P(\omega_1) = P(u)$  and  $x_1 = \delta(q_0, uu')$ . Since  $x_1$  has normal label in  $x_{V,1}$ , then neither  $u$  nor  $u'$  contains  $\sigma_p$ . Also, since  $P(\omega_1) = P(\omega_1 s_{uo}) = P(s_1) = P(t_1)$ , then  $P(u) = P(t_1)$ . If there is a cycle formed by  $\{x_{V,1}, \dots, x_{V,m}\}$  and  $\sigma_1 \dots \sigma_m \in \Sigma^*$ , then there is a corresponding cycle in  $G$  formed by normal states in  $x_{V,i}$  for  $i = 1, \dots, m$  and a subsequence  $\sigma'_1 \dots \sigma'_{m'} \in \Sigma^*$  where  $m' \leq m$  is a positive integer. Thus,  $x_1 = \delta(q_0, uu'(\sigma'_1 \dots \sigma'_{m'})^k)$  for some integer  $k \geq n$  and  $u'(\sigma'_1 \dots \sigma'_{m'})^k$  does not contain  $\sigma_p$ . Pick  $v = u'(\sigma'_1 \dots \sigma'_{m'})^k \in L/u$ . By the above discussion, neither  $u$  nor  $v$  contains  $\sigma_p$ . Thus, there exist  $u$  and  $v \in L/u$  such that  $P(u) = P(t_1)$ ,  $\sigma_p \notin u$ ,  $\|v\| \geq n$  and  $\sigma_p \notin v$ . This is a violation of the condition,  $\mathbf{P}$ . Thus,  $\sigma_p$  is not predictable in  $L$ . This is a contradiction. This completes the first part of the proof.

( $\Leftarrow$ ): We prove that if for all  $q_V \in F_V$  the only cycles in  $Ac(V_G, q_V)$  are cycles of certain verifier states, then  $\sigma_p$  is predictable in  $L$ . Without loss of generality, pick any  $s \in \Psi(\sigma_p, L)$  such that  $s = s_1 \sigma_p$  where  $s_1$  does not contain  $\sigma_p$ . (It suffices for the sake of predictability to consider the first occurrence of  $\sigma_p$  along any string.) Let  $x = \delta(q_0, s_1)$  and  $y = \delta(x, \sigma_p)$ . Then, there exist  $x_V = [(x, N), (x', l'_x)]$  and  $y_V = [(y, F), (y', l'_y)]$  in  $Q_V$  such that  $y_V \in \delta_V(x_V, \sigma_p)$  where  $x', y' \in Q$  and  $l'_x, l'_y \in \{N, F1\}$ . The verifier state  $x_V$  is either normal or uncertain. Also,  $y_V$  is either uncertain or certain. We now consider the following two cases: (i)  $x_V \in Q_V^N$  and (ii)  $x_V \in Q_V^U$ .

Case (i). Since  $x_V \in Q_V^N$  and  $y_V \notin Q_V^N$ , then  $x_V \in F_V$ . We choose  $t = s_1$ . Pick  $u \in L$  such that  $P(u) = P(t) =$

$P(s_1)$  and  $\sigma_p \notin u$ . Let  $z = \delta(q_0, u) \in Q$ . Since  $\sigma_p \notin u$ ,  $z$  in a verifier state can only have label  $N$ . We wish to show that all the verifier states  $q_V$  that contain  $(z, N)$  are in  $F_V$ . If we show that  $q_V \in F_V$ , then the proof will be complete. This is because the only cycles in  $Ac(V_G, q_V)$  are of certain states and thus, for all  $u$  and  $v \in L/u$ ,  $v$  contains  $\sigma_p$ . Hence,  $\sigma_p$  is predictable.

We now prove that for any  $q_V$  that contains  $(z, N)$  where  $z = \delta(q_0, u) \in Q$ ,  $q_V \in F_V$ . We now consider two cases: (1)  $(x, N)$  is also in  $q_V$  and (2)  $(x, N)$  is not in  $q_V$ .

*Case (i.1).* The verifier state is of the form  $q_V = [(z, N), (x, N)]$ . The event  $\sigma_p$  is feasible from  $x$  and  $y = \delta(x, \sigma_p)$ . By Lemma 10, there exists a verifier state  $[(x, N), (x, N)] \in Q_V$  and  $[(x, N), (x, N)] \in F'_V$  since  $\sigma_p$  is feasible from  $x$ . Thus, by definition of  $F_V$ ,  $q_V \in F_V$ .

*Case (i.2).* The verifier state is of the form  $q_V = [(z, N), (z', l'_z)]$ . By Lemma 10, since  $P(t) = P(u)$ , there exists a state  $[(z, N), (x, N)]$  and by the proof in *Case (i.1)*,  $[(z, N), (x, N)] \in Q_V$ . Thus, by definition of  $F_V$ ,  $q_V \in F_V$ .

*Case (ii).* If  $x_V \in Q_V^U$ , i.e.,  $x_V$  is not normal, then we wish to find a normal verifier state in  $F_V$  from which  $x_V$  is reached. By Lemma 12, there exists a verifier state  $w_V$  reachable from the initial verifier state,  $x_V$  is accessible from  $w_V$ , and  $w_V$  is in  $F_V$ . Then, since  $F_V$  consists of normal verifier states,  $w_V$  is in  $Q_V^N$ . Thus, the proof of *Case (ii)* reduces to the proof of *Case (i)* in which we substitute  $w_V \in Q_V^N$  for  $x_V \in Q_V^N$ . This completes the second part of the proof.  $\square$

Consider  $G_1$  and  $G_2$  in Fig. 1 and Fig. 2, and the corresponding verifiers  $V_{G_1}$  and  $V_{G_2}$  shown in Fig. 5 and Fig. 6, respectively, where  $\sigma_p = \{f\}$ . The set  $F_V = F_V = \{[(5, N), (5, N)]\}$  in  $V_{G_1}$ . The accessible FSA from  $[(5, N), (5, N)]$  contains only one cycle which is a cycle of certain verifier states. Thus,  $f$  is predictable in  $L_1$ . In  $V_{G_2}$ ,  $F_V = \{[(3, N), (3, N)], [(4, N), (3, N)], [(4, N), (4, N)]\}$ . The accessible FSA from the verifier state  $[(4, N), (4, N)]$  contains a cycle of normal verifier states. Thus,  $V_{G_2}$  does not satisfy the necessary and sufficient condition in Theorem 13 and  $f$  is not predictable in  $L_2$ .

## 5 Illustrative Examples

### 5.1 A HVAC System

We present an example of an elementary HVAC (see [20] for more details). The tools UMDES and DESUMA are used to analyze this example. We consider the valve, pump, load, fan, boiler and controller models shown in Figs. 7, 9, 8, 10, 11 and 12, respectively. The valve automaton has four states, *valve\_open* ( $VO$ ), *valve close*

( $VC$ ), *stuck open* ( $SO$ ), and *stuck closed* ( $SC$ ) and six events, *open*, *close*, *stuck\_open*, and *stuck\_closed*. The initial state of the valve is  $VC$ . The pump automaton has two states, *pump on* ( $PO$ ) and *pump off* ( $POFF$ ) and two events, *start* and *stop*. The initial state of the pump is  $POFF$ . The fan automaton has two states, *fan on* ( $F1$ ) and *fan off* ( $F2$ ) where  $em F2$  is the initial state and two events, *fanon* and *fanoff*. The boiler automaton has two states, *boiler on* ( $B1$ ) and *boiler off* ( $B2$ ) where  $B2$  is the initial state and two events, *boileron* and *boileroff*.

The load automaton has 3 states,  $L0$ ,  $L1$ , and  $L2$  and three events, *decrease*, *increase*, and *power\_off*. State  $L0$  represents an unknown load. The states  $L1$  and  $L2$  represent the absence and presence, respectively, of a heating load. The set point increases when there is a demand on the heating system, hence, the state transitions from  $L0$  to  $L2$ . Similarly, the set point decreases when there is no load in the system, hence, the state transitions from  $L0$  to  $L1$ . The initial state is  $L0$ .

The controller issues several sequences of commands. In the normal mode of operation, when there is a load in the system, the controller issues *open valve*, *start*, and *boiler on*, consecutively. When there is no load, it issues *close valve*, *stop*, *boiler off*. When the controller fails off (*failoff*), it does not issue any of the previous commands. On the other hand, when the controller fails on (*failon*), it assumes there is a load in the system and it issues *open* (valve), *start* (pump), and *boileron*, consecutively, no matter if there is a load or not.

The system  $G$  is the parallel composition of the valve, pump, load, fan, boiler and controller. The unobservable events are *stuck\_open*, *stuck\_closed*, *failoff* and *failon*. The FSA  $G$  has 186 states and 439 transitions. We investigate the predictability of two different events in the language generated by  $G$ ,  $\mathcal{L}(G)$ : (i)  $\sigma_p = stop$  and (ii)  $\sigma_p = close$ . Since the corresponding diagnosers or verifiers are too large to show, we only present the results of the analysis.

*Case (i) [ $\sigma_p = stop$ ]* The event  $\sigma_p = stop$  is not predictable in  $\mathcal{L}(G)$ . This was confirmed using the diagnoser-based necessary and sufficient condition. An examination of this diagnoser reveals the following violation of predictability due to state in  $F_D$  followed by a cycle of uncertain states: The string *fanon increase open start boileron decrease stop* ends with *stop*; however, there is no prefix of that string such that its projections and the continuations of the projections are distinct enough to predict the occurrence of event *stop*.

*Case (ii) [ $\sigma_p = close$ ]* The event  $\sigma_p = close$  is predictable in  $\mathcal{L}(G)$ , as confirmed by the diagnoser-based test. The test reveals that all the strings that end with *close* have a prefix whose projection contains the event *stop*. Thus, upon observation of the event *stop*, the di-

agnoser can predict that event *close* will for sure occur in the future evolution of the system.

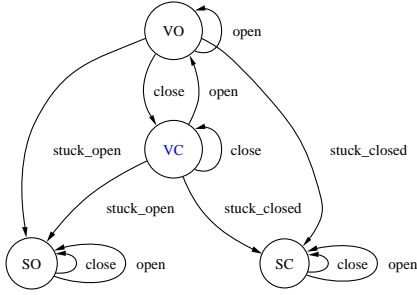


Fig. 7. Valve.

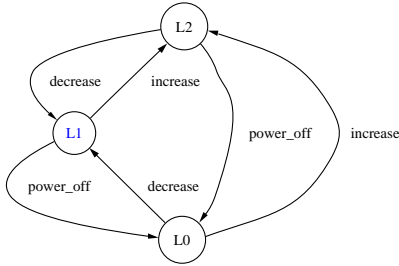


Fig. 8. Load.

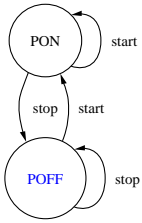


Fig. 9. Pump.

Fig. 10. Fan.

Fig. 11. Boiler.

## 5.2 Intrusion Prediction

References [9,10] present applications of the theory of diagnosability of DES to intrusion detection problems in computer systems. Automata models are built based on logs of operating-system-level events, and the objective is to diagnose events or sequences of events corresponding to potential intrusions. The results on predictability presented in this paper could also be applied in this area to perform intrusion *prediction*. The same modeling strategies as in [9,10] would be used, but a prediction problem would be solved instead of a diagnosis one.

We consider an illustrative example related to intrusions in networked systems. In [14], the authors develop a tool called *BackTracker* that builds *dependency graphs* to identify the sequences of operating-system(OS)-level events that lead to an intrusion. Then, an administrator may analyze these sequences of events to quickly identify vulnerabilities in the system. However, the dependency graphs generated by *BackTracker* may contain too many

Table 1  
The sample event log.

Time	Log
0	process_O reads creates_A
1	process_A creates process_B
2	process_B writes file_1
3	process_B writes file_2
4	process_A writes file_0
5	file_0 is locked (busy)
6	process_A creates process_C
7	process_C reads file_1
8	process_C writes file_3
9	process_D reads file_3
10	process_C reads file_2
11	process_D is busy

events for an administrator to analyze. Thus, in [14], the authors apply some filtering rules to reduce the size of the dependency graphs. That is, the dependency graph is filtered using a set of observable events. The administrator may then analyze the smaller graph for vulnerabilities in the system to a known or a possible intrusion. This can involve considering an event to be a known or possible intrusion and then verify the predictability of that event (intrusion) in the dependency graph built from the event logs. Our objective in this example is two-fold: (i) provide a method based on the results developed in Section 4 to check for predictability of the intrusion and (ii) show that the set of observable (filtering) events plays a role in predicting the occurrence of an intrusion modeled as an event. For these purposes, we consider the sample event log in Table 1. We build the dependency graph in the form of the nondeterministic FSA  $G$  shown in Fig. 13. The event set is  $\Sigma = \{\text{busy, create, read, write}\}$ . We wish to predict occurrences of event read in  $G$ .

First, suppose that the set of observable events is  $\Sigma_{o,1} = \{\text{create, busy}\}$ . The diagnoser built from  $G$  for  $\Sigma_{o,1}$  and  $\sigma_p = \text{read}$  is shown in Fig. 14. The set  $F_D$ , the set of normal diagnoser states with an uncertain or a certain diagnoser state as an immediate successor, is  $F_D = \{\{process\_CN, process\_BN\}\}$ . The accessible part  $Ac(G, \{process\_CN, process\_BN\})$  contains only one cycle with a single certain diagnoser state. Thus, the occurrences of read are predictable.

Second, suppose that the set of observable events is  $\Sigma_2 = \{\text{write, busy}\}$ . The diagnoser built from  $G$  for  $\Sigma_{o,2}$  and  $\sigma_p = \{\text{read}\}$  is shown in Fig. 15. In this case,  $F_D = \{\{file\_3N, file\_2N, file\_1N, file\_0N\}\}$ . The accessible part  $Ac(G, \{file\_3N, file\_2N, file\_1N, file\_0N\})$  contains two cycles. One of these cycles is formed by a single certain diagnoser state  $\{process\_DF1\}$ . However, the other cycle is formed by a single uncertain diagnoser state  $\{process\_DF1, file\_0N\}$ . Thus, the accessible part contains a cycle of uncertain diagnoser states. This violates the necessary and sufficient condition for predictability of occurrences of event read in  $G$ .

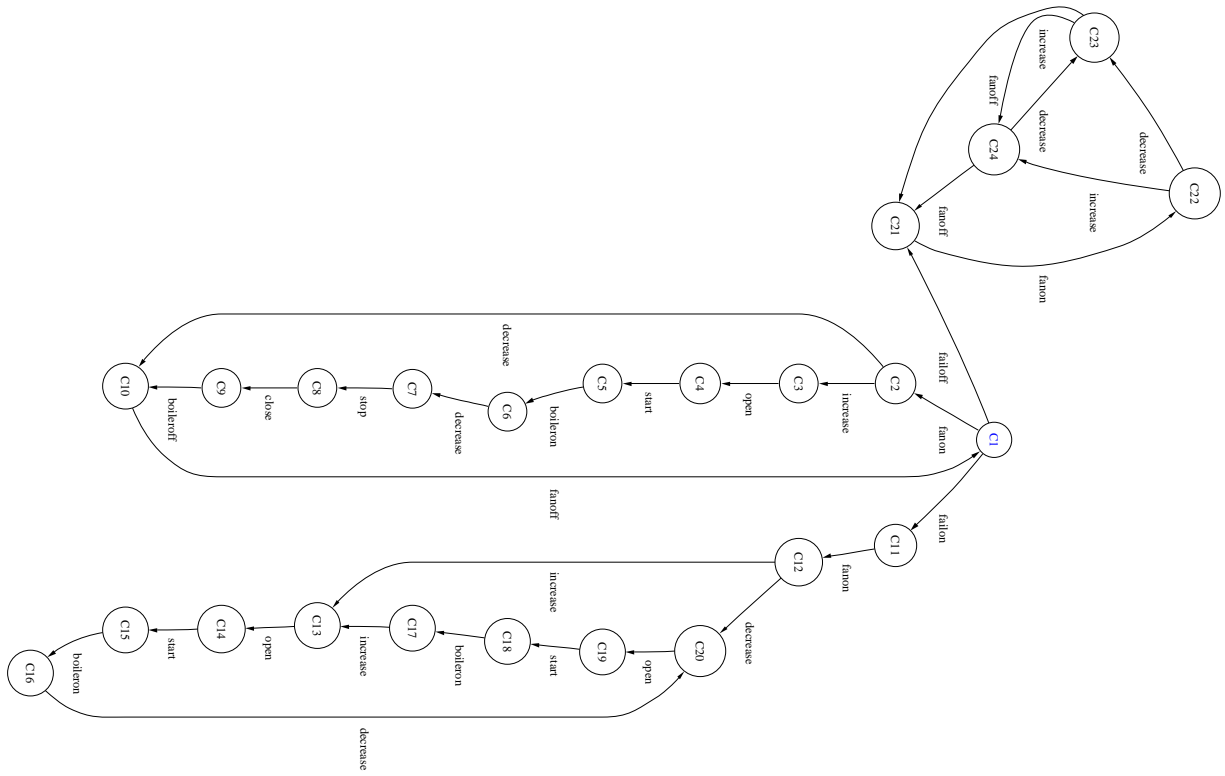


Fig. 12. Controller.

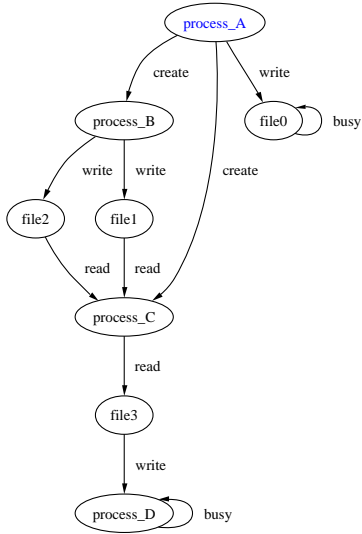


Fig. 13. FSA  $G$  built from the sample event log.

## 6 Conclusion

We have defined the new property of predictability of the occurrence of a significant event, which could be observable or unobservable, based on the current record of observable events. We have presented necessary and

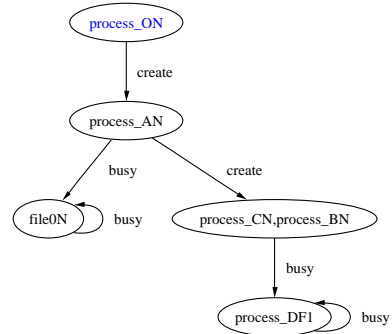


Fig. 14.  $D_G$  for  $\Sigma_1 = \{\text{create, busy}\}$ .

sufficient conditions for predictability in the case of systems modeled by regular languages. The first set of conditions is based on the familiar diagnoser automata used in the study of the property of diagnosability of DES. The structure of diagnosers allows to characterize completely predictability in terms of an accessibility test. Diagnosers can also be used online to issue the prediction decisions for predictable systems. An alternate offline test of polynomial-time complexity (in the number of system states) based on verifiers is also presented. Again, it is possible to exploit the structure of verifier automata, initially proposed to study diagnosability, and obtain an effective test for predictability.

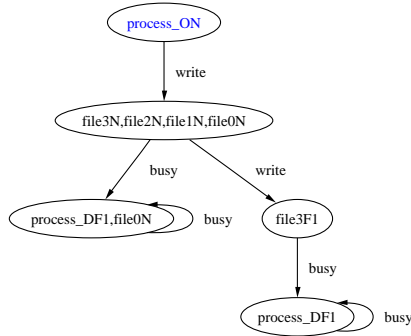


Fig. 15.  $D_G$  for  $\Sigma_2 = \{\text{write}, \text{busy}\}$ .

Many options for future research on predictability are open. One is the notion of prediction of patterns of events, as opposed to single events. Another is the consideration of modular approaches to predictability. If the system model  $G$  is given as the parallel composition of a set of component automata, then the goal would be to exploit this structure to improve the computational efficiency of the test for predictability. Finally, there is a large body of research on diagnosability of systems with decentralized information structures; it would be interesting to study predictability in the context of similar architectures. One is the notion of prediction of patterns of events, as opposed to single events [13].

## Acknowledgements

It is a pleasure to acknowledge H. Marchand, whose insightful comments helped to improve the paper. The comments of the reviewers are also acknowledged.

## References

- [1] S. R. Buss, C. Papadimitriou, and J. Tsitsiklis. On the predictability of coupled automata: an allegory about chaos. *Complex Systems*, 5:525–539, 1991.
- [2] X. Cao. The predictability of discrete event systems. *IEEE Trans. Automatic Control*, 34(11):1168–1171, November 1989.
- [3] C. Chase and P. J. Ramadge. Predictability of a class of one-dimensional supervised systems. In *Proceedings of the Fifth IEEE International Symposium on Intelligent Control*, September 1990.
- [4] S. K. Das, F. Sarkar, K. Basu, and S. Madhavapeddy. Parallel discrete event simulation in star networks with application to telecommunications. In *MASCOTS '95: Proceedings of the 3rd International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, pages 66–71, Washington, DC, USA, 1995. IEEE Computer Society.
- [5] P. Declerck. Predictability and control synthesis in time deviant graphs. In *Workshop on Discrete-Event Systems*, Cagliari, Italy, August 1998.
- [6] H.K. Fadel and L.E. Holloway. Using SPC and template monitoring method for fault detection and prediction in discrete event manufacturing systems. In *Proceedings of the 1999 IEEE International Symposium on Intelligent Control/Intelligent Systems and Semiotics*, pages 150 – 155, September 1999.
- [7] P. H. Feiler, B. Lewis, and S. Vestal. Improving predictability in embedded real-time systems, December 2000.
- [8] S. Genc. On diagnosis and predictability of partially-observed discrete-event systems. Ph.D. Thesis. University of Michigan, Electrical Engineering: Systems. April 2006.
- [9] S. Genc. Formal methods for intrusion detection of windows nt attacks. In *3rd Annual Symposium on Information Assurance (ASIA08) and 11th Annual NYS Cyber Security Conference*, Albany, NY, June 2008.
- [10] S. Genc and S. Lafortune. Diagnosis of patterns in partially-observed discrete-event systems. In *45th IEEE Conference on Decision and Control*, San Diego, CA, 2006.
- [11] S. Genc and S. Lafortune. Predictability in discrete-event systems under partial observation. In *6th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*, Beijing, PR China, 2006.
- [12] Q. He, M. Ammar, G. Riley, and R. Fujimoto. Exploiting the predictability of tcp's steady-state behavior to speed up network simulation. In *10th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems*, pages 101–108, 2002.
- [13] T. Jeron, H. Marchand, S. Genc, and S. Lafortune. Predictability of sequence patterns in discrete event systems. In *17th IFAC World Conference*, Seoul, South Korea, 2008.
- [14] S. T. King and P. M. Chen. Backtracking intrusions. *ACM Trans. Comput. Syst.*, 23(1):51–76, February 2005.
- [15] S. Lafortune. Umdes-lib software library. <http://www.eecs.umich.edu/umdes/toolboxes.html>.
- [16] M. Musolesi and C. Mascolo. Evaluating context information predictability for autonomic communication. In *American Control Conference*, Minneapolis, Minnesota USA, June 2006.
- [17] L. Ricker, S. Lafortune, and S. Genc. Desuma: A tool integrating giddes and umdes. In *Software Tools, 8th International Workshop on Discrete-Event Systems*, July 2006.
- [18] M. Sampath, S. Lafortune, and D. Teneketzis. Active diagnosis of discrete event systems. *IEEE Trans. Automatic Control*, 43(7):908–929, July 1998.
- [19] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis. Diagnosability of discrete event systems. *IEEE Trans. Automatic Control*, 40(9):1555–1575, September 1995.
- [20] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis. Failure diagnosis using discrete event models. *IEEE Trans. Control Systems Technology*, 4(2):105–124, March 1996.
- [21] J. Shengbing and R. Kumar. Failure diagnosis of discrete-event systems with linear-time temporal logic specifications. *IEEE Trans. Automatic Control*, 49(6):934 – 945, June 2004.
- [22] T. Yoo and S. Lafortune. Polynomial-time verification of diagnosability of partially-observed discrete-event systems. *IEEE Transactions of Automatic Control*, 47(9):1491–1495, 2002.
- [23] F. Zhao, X. Koutsoukos, H. Haussecker, J. Reich, and P. Cheung. Monitoring and fault diagnosis of hybrid systems. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 35 (6):1225–1240, December 2005.