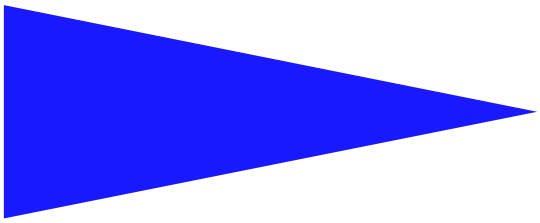
I R I S A
INSTITUT DE RECHERCHE EN INFORMATIQUE ET SYSTÈMES ALÉATOIRES

PUBLICATION
INTERNE
N° 1834



# PREDICTABILITY OF SEQUENCE PATTERNS IN DISCRETE EVENT SYSTEMS

THIERRY JÉRON, HERVÉ MARCHAND, SAHIKA GENC, STÉPHANE LAFORTUNE

ISSN 1166-8687

IRISA
CAMPUS UNIVERSITAIRE DE BEAULIEU - 35042 RENNES CEDEX - FRANCE

IRISA
Tél. : (33) 02 99 84 71 00 – Fax : (33) 02 99 84 71 71
http://www.irisa.fr

# Predictability of Sequence Patterns in Discrete Event Systems*

Thierry Jéron, Hervé Marchand, Sahika Genc** , Stéphane Lafortune***

Systèmes communicants
Projet VerTeCs

Publication interne n ° 1834 — March 2007 — 17 pages

**Abstract:** The problem of predicting the occurrences of a pattern in a partially-observed discrete-event system is studied. The system is modeled by a labeled transition system. The pattern is a set of event sequences modeled by a finite-state automaton. The occurrences of the pattern are predictable if it is possible to infer about any occurrence of the pattern before the pattern is completely executed by the system. An off-line algorithm to verify the property of predictability is presented. The verification is polynomial in the number of states of the system. An on-line algorithm to track the execution of the pattern during the operation of the system is also presented. This algorithm is based on the use of a diagnoser automaton. The results are illustrated using an example from computer systems.

**Key-words:** Labelled Transition Systems, Sequence patterns, Prediction

*(Résumé : tsvp)*

* The research of S. Lafortune is supported in part by NSF grants CCR-0325571 and ECCS-0624821. The research of T. Jéron and H. Marchand is supported by RNTL Politess and ACI Potestat.

** Sahika Genc is with General Electric Global Research Center, 1 Research Circle, Niskayuna, NY 12309 gencs@ge.com

*** Stéphane Lafortune is with the University of Michigan, Ann Arbor, MI, USA stephane@eecs.umich.edu

Centre National de la Recherche Scientifique (UMR 6074) Université de Rennes 1 – Insa de Rennes

Institut National de Recherche en Informatique et en Automatique – unité de recherche de Rennes

# Prédiction de motifs de séquences pour des systèmes à événements discrets

**Résumé :** Dans ce rapport, nous nous intéressons à la prédiction d'occurrences de motifs de pannes dans des systèmes à événements discrets. Le système est modélisé par des systèmes de transitions finis. Le motif de pannes est donné par un ensemble de séquences. Ce motif est prédictif si il et possible d'inférer l'occurrence de ce motif avant son exécution réelle par le système. Nous présentons un algorithme hors ligne de vérification de la prédictabilité, avec une complexité polynomiale en le nombre d'états du système. Par ailleurs, nous montrons comment construire un "diagnostiqueur" en charge de la surveillance de ce motif durant l'exécution du système.

**Mots clés :** systèmes de transitions finis, motifs de pannes, Prediction

# 1 Introduction

Monitoring and fault diagnosis of dynamic systems modeled as discrete event systems has received considerable attention in the literature in the last decade. Several methodologies have been developed for different types of diagnosis problems. The methodology upon which this paper is based is the Diagnoser Approach; see, e.g., [17] for a tutorial overview of this approach. This paper is different from prior works on the Diagnoser Approach because it considers the problem of *predicting sequence patterns* in the behavior of a partially-observed discrete-event system (DES). Partially-observed DES have observable events (e.g., sensor readings, changes in the sensor readings, etc.) and unobservable events, i.e., events that are not directly recorded by the sensors attached to the system. The sequence pattern, or simply pattern, to be predicted consists of ordered occurrences of one or more significant observable or unobservable events. The consideration of sequence patterns is a more general form of the prediction of single (fault) events, a problem that has received attention recently in the DES literature. If it is possible to predict the occurrences of a pattern that describes event sequences that lead or cause the system to fail or malfunction, then the system operator can be warned in time to halt the system, take preventative measures, or otherwise to reconfigure the system.

In this paper, for the sake of generality, the system is modeled as a Labeled Transition System (LTS) and the pattern as a regular language. The problem of prediction of patterns in DES as considered here is principally based on two prior studies: (i) the problem of *predictability* of single event occurrences in partially-observed DES studied in [10] and (ii) the problem of *diagnosis* of sequence patterns in DES studied in [14] and [9]. The notion of predictability considered in this paper was first introduced in [10] in the case of single events. This notion of predictablility is different from prior works on other notions of predictability in [2, 1, 13, 7]. In [2], the prediction problem is based on the properties of a special type of projection between two languages (sets of trajectories). This is is much more general than our objective. The prediction problem studied in [7] considers the *likelyhood* of a fault happening in the future evolution of the system and it is possible that false *fault prediction warnings* are issued. In our work, if the occurrences of an event pattern is predictable in a language, then it is *certain* that the event pattern will occur and no false positives are issued. In in [1], the state prediction of coupled automata studied is formulated as computing the state vector of $n$ identical automata after $T$ steps in the operation of the system. In our case, the interest is on a single automaton and event prediction, not state, under partial observation. Other references on predictability in DES or discrete event simulation systems are [5, 3, 6, 18, 11, 8]. The results presented in this paper are related to the work of Jiang and Kumar [13] who considered the notion of *inevitability*. Indeed, in [13], the authors are interested in diagnosing inevitability, thus in detecting that a pattern will be inevitably satisfied within a bounded number of observations *after* this inevitability (meaning that the diagnosis of the pattern may actually occur after its occurrence). In contrast, in the context of this paper, predictability means detecting inevitability *strictly before* its occurrence.

In the following sections, we develop a theory for the predictability of pattern occurrences. In Section 2, we define the mathematical terminology and notions used throughout the paper. In Section 3, we specify the notions of pattern and diagnosability considered in this paper. We then define the property of predictability of occurrences of a sequence pattern in Section 4. This notion subsumes and extends the prior notion of predictability in [10]. In Section 5, we propose an off-line polynomial -time algorithm for the verification of predictability. In Section 6, we present an algorithm for constructing a special type of diagnoser for the purpose of on-line prediction of a sequence pattern ansd emphasize some properties of this diagnoser.

## 2 System Model and Notations

Let $\Sigma$ be a finite alphabet 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 $\Sigma^*$. Any subset of $\Sigma^*$ is called a *language* over $\Sigma$. Let $L$ be a language over $\Sigma$. Given a string $s \in L$, $L/s$ is called the *post-language* of $L$ after $s$ and defined as $L/s = \{t \in \Sigma^* \mid s.t \in L\}$.

The system in which the sequence pattern is to be predicted is modeled as an LTS. The formal definition of an LTS is as follows.

**Definition 1 (LTS)** *An LTS over $\Sigma$ is defined by a 4-tuple $M = (Q_M, \Sigma, \rightarrow_M, q^0_M)$ where $Q_M$ is a finite set of states, $\Sigma$ is the set of* events *of $M$, $q^0_M \in Q_M$ is the initial state, and $\rightarrow_M \subseteq Q_M \times \Sigma \times Q_M$ is* the partial transition relation. ◇

In the remainder of this section, we consider a given LTS $M = (Q_{\mathrm{M}}, \Sigma, \rightarrow_{\mathrm{M}}, q^0_{\mathrm{M}})$.

- We write $q \xrightarrow{\sigma}_{\mathrm{M}} q'$ if $(q, \sigma, q') \in \rightarrow_{\mathrm{M}}$. We extend $\rightarrow_{\mathrm{M}}$ to arbitrary sequences by setting : $q \xrightarrow{\varepsilon}_{\mathrm{M}} q$ for all states $q$, and $q \xrightarrow{s\sigma}_{\mathrm{M}} q'$ whenever $q \xrightarrow{s}_{\mathrm{M}} q''$ and $q'' \xrightarrow{\sigma}_{\mathrm{M}} q'$, for some $q'' \in Q_{\mathrm{M}}$.
- We write $q \xrightarrow{s}_{\mathrm{M}}$ if $\exists q' \in Q_{\mathrm{M}}, q \xrightarrow{s}_{\mathrm{M}} q'$. We denote by $\mathcal{L}(M) = \{s \in \Sigma^* \mid q^0_{\mathrm{M}} \xrightarrow{s}_{\mathrm{M}}\}$ the language generated by $M$, which corresponds to the set of trajectories that the LTS $M$ can execute.
- The *event set* of a state $q \in Q_{\mathrm{M}}$ is $\Sigma(q) \triangleq \{\sigma \in \Sigma \mid q \xrightarrow{\sigma}_{\mathrm{M}}\}$. $M$ is *live* if $\Sigma(q) \neq \emptyset$, for each $q \in Q_{\mathrm{M}}$.
- A state $q$ is *reachable* if $\exists s \in \Sigma^*, q^0_{\mathrm{M}} \xrightarrow{s}_{\mathrm{M}} q$.
- We set $\Delta_{\mathrm{M}}(q, s) \triangleq \{q' \in Q_{\mathrm{M}} \mid q \xrightarrow{s}_{\mathrm{M}} q'\}$. By a slight abuse of notation, for any language $L \subseteq \Sigma^*$, $\Delta_{\mathrm{M}}(q, L) \triangleq \bigcup_{s \in L} \Delta_{\mathrm{M}}(q, s)$ and for any $Q' \subseteq Q_{\mathrm{M}}$, $\Delta_{\mathrm{M}}(Q', L) = \bigcup_{q \in Q'} \Delta_{\mathrm{M}}(q, L)$.
- A subset $Q' \subseteq Q_{\mathrm{M}}$ is said *stable* whenever $\Delta_{\mathrm{M}}(Q', \Sigma) \subseteq Q'$. $M$ is *complete* whenever $\forall q \in Q_{\mathrm{M}}, \Sigma(q) = \Sigma$.
- We say that $M$ is *deterministic* if, whenever $q \xrightarrow{\sigma}_{\mathrm{M}} q'$ and $q \xrightarrow{\sigma}_{\mathrm{M}} q''$, then $q' = q''$, for all $q \in Q_{\mathrm{M}}$ and all $\sigma \in \Sigma$.

We denote the set of final states by $F_{\mathrm{M}} \subseteq Q_{\mathrm{M}}$. The notions above are extended in this setting by letting the language recognized in $F_{\mathrm{M}}$ be

$$\mathcal{L}_{F_{\mathrm{M}}}(M) = \{s \in \Sigma^* \mid \Delta_{\mathrm{M}}(q^0_{\mathrm{M}}, s) \subseteq F_{\mathrm{M}}\}.$$

We now define the synchronous product of two LTS.

**Definition 2** *Let $M^i = (Q^i, \Sigma, \rightarrow_{M^i}, q^0_{M^i})$, $i = 1, 2$, be two LTS. Their* synchronous product *is $M^1 \times M^2 \triangleq (Q^1 \times Q^2, \Sigma, \rightarrow_{M^1 \times M^2}, (q^0_{M^1}, q^0_{M^2}))$, where $\rightarrow_{M^1 \times M^2} \subseteq (Q^1 \times Q^2) \times \Sigma \times (Q^1 \times Q^2)$ satisfies $(q^1, q^2) \xrightarrow{\sigma}_{M^1 \times M^2} (q'^1, q'^2)$ whenever $q^1 \xrightarrow{\sigma}_{M^1} q'^1$ and $q^2 \xrightarrow{\sigma}_{M^2} q'^2$.* ◇

Clearly $\mathcal{L}(M^1 \times M^2) = \mathcal{L}(M^1) \cap \mathcal{L}(M^2)$ and for $F_i \subseteq Q^i, i = 1, 2$, we also have $\mathcal{L}_{F_1 \times F_2}(M^1 \times M^2) = \mathcal{L}_{F_1}(M^1) \cap \mathcal{L}_{F_2}(M^2)$. Also, if sets $F_i$ are stable in $M^i$, $F_1 \times F_2$ is stable in $M^1 \times M^2$.

Given a set of states $E \subseteq Q_{\mathrm{M}}$ of $M$, we define the following sets

$$Pre^{\forall}_{\mathrm{M}}(E) = \{q \in Q \mid \forall \sigma \in \Sigma, \Delta_{\mathrm{M}}(q, \sigma) \subseteq E\}$$

$$Pre^{\exists}_{\mathrm{M}}(E) = \{q \in Q \mid \exists \sigma \in \Sigma, \Delta_{\mathrm{M}}(q, \sigma) \cap E \neq \emptyset\}$$

In words, states belonging to $Pre^{\forall}_{\mathrm{M}}(E)$ are states such that all immediate successors belong to $E$, while states belonging to $Pre^{\exists}_{\mathrm{M}}(E)$ are states such that at least one immediate successor belongs to $E$.

Given a live LTS $M$, let $Inev_M(E)$ be the set of states that inevitably lead to a set $E$ in a finite number of steps. This set is given by:

$$\begin{aligned} Inev_{\mathrm{M}}(E) \quad = \quad & \{q \in Q \mid \exists n \geq 0, s.t. \forall s \in \Sigma^*, q \xrightarrow{s}_{\mathrm{M}} \wedge \\ & \|s\| \geq n \Rightarrow \ \Delta_{\mathrm{M}}(q,s) \subseteq E\} \end{aligned}$$

$Inev_M(E)$ can also be characterized by a least fixpoint (lfp) as follows[1]:

$$Inev_{\mathrm{M}}(E) = lfp(\lambda X.E \cup Pre^{\forall}_{\mathrm{M}}(X))$$

The set of events $\Sigma$ is partitioned into $\Sigma_o$ and $\Sigma_{uo}$

$$\Sigma = \Sigma_o \cup \Sigma_{uo}, \text{ and } \Sigma_o \cap \Sigma_{uo} = \emptyset,$$

where, as usual, $\Sigma_o$ is the set of *observable* events while $\Sigma_{uo}$ is the set of *unobservable* events. In this paper, the elements of $\Sigma_o^*$ will be denoted by $\mu, \mu'$. We say that $M$ is $\Sigma_o$-*live* if $\forall q \in Q, \exists s \in \Sigma^*.\Sigma_o, q \xrightarrow{s}$, meaning that there is no terminal loop of unobservable events.

$P : \Sigma^* \rightarrow \Sigma_o^*$ denotes the natural *projection* of trajectories onto $\Sigma_o^*$. This is extended to any language $L \subseteq \Sigma^*$ by letting $P(L) = \{P(s) \mid s \in L\}$. We adopt the terminology *traces* for *observable* trajectories; note that while standard in computer science, this notation is not standard in the DES literature. Given an LTS $M$, the set of traces of $M$ is thus given by $P(\mathcal{L}(M))$. The inverse projection of a language $L \subseteq \Sigma_o^*$ is defined by $P^{-1}(L) = \{s \in \Sigma^* \mid P(s) \in L\}$.

Given a trace $\mu$ of $M$, we define $[\![\mu]\!]$ as the set of trajectories *compatible* with the trace $\mu$:

$$\begin{aligned} [\![\mu]\!] \quad \triangleq \quad & P^{-1}(\mu) \cap \mathcal{L}(M) \cap \Sigma^*\Sigma_o \text{ if } \mu \neq \epsilon \\ & \epsilon \text{ otherwise.} \end{aligned}$$

This means that (except for the empty trace), trajectories compatible with a trace $\mu$ are trajectories of $M$ ending with an observable event and having trace $\mu$. This is consistent with an on-line diagnosis where verdicts consider trajectories until the last observation, and not subsequent unobservable events.

Next, we introduce the Unobservable-Closure $\textsc{Uc}(M)$ of an LTS $M$, in order to abstract out unobservable events according to the semantic $[\![.]\!]$.

**Definition 3** *For an LTS $M = (Q_M, \Sigma, \rightarrow, q^0_M)$, the* Unobservable-Closure of $M$ *is* $\textsc{Uc}(M) = (Q_M, \Sigma_o, \rightarrow_{Uc}, q^0_M)$ *where for any $q, q' \in Q_M$, $\sigma \in \Sigma_o$, $q \xrightarrow{\sigma}_{Uc} q'$ in $\textsc{Uc}(M)$ whenever there exists $s \in \Sigma^*_{uo}$ such that $q \xrightarrow{s\sigma}_M q'$ in $M$.* ◇

We get $\mathcal{L}(\textsc{Uc}(M)) = P(\mathcal{L}(M))$ and $\mathcal{L}_{F_{\mathrm{M}}}(\textsc{Uc}(M)) = P(\mathcal{L}_{F_{\mathrm{M}}}(M))$

Determinization of an LTS $M$ defined on the alphabet $\Sigma = \Sigma_o \cup \Sigma_{uo}$ produces an LTS $Det(M)$ with actions in $\Sigma_o$, and deterministic on $\Sigma_o$, with same traces as the $M$.

**Definition 4** *Let $M = (Q_M, \Sigma, \rightarrow_M, q^0_M)$ be an LTS with $\Sigma = \Sigma_{uo} \cup \Sigma_o$. The* determinization of $M$ *is the LTS $Det(M) = (\mathcal{X}, \Sigma_o, \rightarrow_d, X^0)$ where $\mathcal{X} = 2^{Q_M}$ (the set of subsets of $Q_M$ called* macro-states*), $X^0 = \{q^0_M\}$ and $\rightarrow_d = \{(X, \sigma, \Delta_M(X, \Sigma^*_{uo}.\sigma)) \mid X \in \mathcal{X}$ and $\sigma \in \Sigma_o\}$.* ◇

Notice that this definition differs from the classical determinization of automata, but is consistent with the above semantic of observations $[\![.]\!]$: the target macro-state $X'$ of a transition $X \xrightarrow{\sigma}_d X'$ is only composed of states $q'$ of $M$ which are targets of sequences of transitions $q \xrightarrow{s.\sigma} q'$ ending with an

[1] We assume the reader is familiar with fixpoint notation.

observable event $\sigma$. As a consequence, $Det(M)$ can be constructed in two steps by first constructing $\textsc{Uc}(M)$ and then applying the classical subset construction used in the usual determinization of automata.

From the definition of $\rightarrow_d$ in $Det(M)$, we infer that $\Delta_{\mathrm{Det(M)}}(X^0, \mu) = \{\Delta_{\mathrm{M}}(q^0_{\mathrm{M}}, [\![\mu]\!])\}$, which means that the macro-state reached from $X^0$ by $\mu$ in $Det(M)$ is composed of the set of states that are reached from $q^0_{\mathrm{M}}$ by trajectories of $[\![\mu]\!]$ in $M$.

Finally, determinization preserves observations, so we have $\mathcal{L}(Det(M)) = P(\mathcal{L}(M))$. Also for $F_{\mathrm{M}} \subseteq Q_{\mathrm{M}}$, we have $\mathcal{L}_{2^{F_{\mathrm{M}}}}(Det(M)) = \{\mu \in \Sigma_o^* \mid [\![\mu]\!] \subseteq \mathcal{L}_{F_{\mathrm{M}}}(M)\}$.

## 3 Patterns and Diagnosability

In this section, we define patterns that describe stable properties, i.e., negation of safety properties; see [14] for further details. We also recall the definition of the property of diagnosability. We assume that the system is modeled by an LTS $G = (Q_{\mathrm{G}}, \Sigma, \rightarrow_{\mathrm{G}}, q^0_{\mathrm{G}})$ that is $\Sigma_o$-live and whose event set $\Sigma$ includes a set of unobservable events $\Sigma_{uo}$ with associated projection operation $P$.

**Definition 5** *A* sequence pattern, *or simply pattern, is a deterministic and complete LTS,* $\Omega = (Q_\Omega, \Sigma, \rightarrow_\Omega, q^0_\Omega)$ *equiped with a distinguished stable subset of states* $F_\Omega \subseteq Q_\Omega$. ◇

As $\Omega$ is complete we get $\mathcal{L}(\Omega) = \Sigma^*$. Also note that the assumption that $F_\Omega$ is stable means that its recognized language is "extension-closed", i.e., it satisfies $\mathcal{L}_{F_\Omega}(\Omega).\Sigma^* = \mathcal{L}_{F_\Omega}(\Omega)$. In other words, $\mathcal{L}_{F_\Omega}(\Omega)$ is a language violating a safety property. In [14], examples of patterns often used in diagnosis are presented, including the cases "occurrence of a single fault event", "occurrence of one of multiple fault events in the same partition", "ordered occurrence of events", "multiple occurrences of the same fault", etc. The same approach is used in this paper in the context of the predictability problem.

Given an LTS $G$ and a pattern $\Omega$, we say that a trajectory $s \in \mathcal{L}(G)$ is recognized by the pattern if $s \in \mathcal{L}_{F_\Omega}(\Omega)$. Diagnosability of an LTS $G$ with respect to a pattern $\Omega$ equipped with $F_\Omega$, is defined as the ability to detect that a trajectory is recognized by $\Omega$ on the basis of the observed projection of the trajectory only, within a bounded number of observations. Formally, we have:

**Definition 6** *An LTS* $G$ *is* $\Omega$-diagnosable *if* $\exists n \in \mathbb{N}$,

$$\forall s \in \mathcal{L}_{F_\Omega}(\Omega) \cap \mathcal{L}(G) \cap \Sigma^*\Sigma_o, \forall t \in \mathcal{L}(G)/s \cap \Sigma^*\Sigma_o,$$
$$\textit{if } \|P(t)\| \geq n \quad \textit{then } [\![P(s.t)]\!] \subseteq \mathcal{L}_{F_\Omega}(\Omega).$$

◇

$\Omega$-diagnosability says that when a trajectory $s$ ending with an observable event is recognized by the pattern $\Omega$, for any extension $t$ with enough observable events, any trajectory $s'$ compatible with the observation $P(s.t)$ is also recognized by $\Omega$. Details can be found in [14].

## 4 Predictability

In this section, we introduce the notion of *predictability of a pattern*. $G$ and $\Omega$ are as defined in the preceding section. Roughly speaking, a pattern is predictable if it is always possible to detect the future recognition of the pattern, strictly before this recognition, only based on the observations. We explain the idea of predictability, and how it differs from diagnosability, in the following example.

**Example 1** *In [16], a method is developed to build a dependency graph from a log file which records the commands issued in an operation system. Dependencies between processes and files are defined based on the nature of the command issued. For example, let a "write" command be issued by the process "Process_A" to access to the file "file_0". If both the process and file are represented by states,*

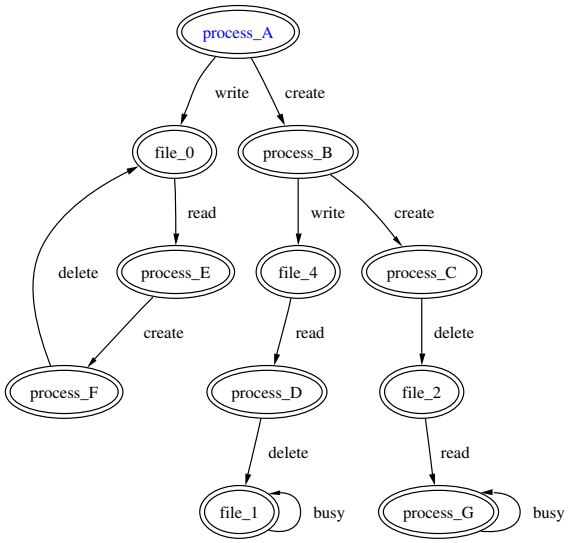



Figure 1: The LTS $G_1$ of Example 1

*then we draw a transition from the state 'Process_A" to the file "file_0" and label the transition with the command "write". Consider the LTS $G_1$ shown in Fig. 1. Suppose that $G_1$ is built using a complete set of dependency relations among given sets of processes and files. The set of events is $\Sigma = \{read,\ write,\ delete,\ busy,\ create\}$. Let $\Sigma_{uo} = \{create, delete\}$ be the set of unobservable events. A significant pattern is the occurrence of delete followed by either read or write, without any create in between. This pattern is of importance because a process is trying to read or write a file which is already deleted This pattern is given in Fig. 2 and denoted by $\Omega$ in this example.*

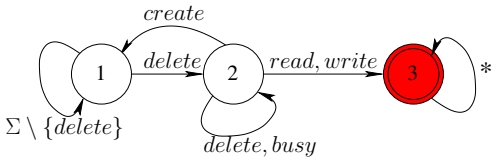



Figure 2: The $\Omega$ pattern of Example 1

*By a simple inspection of $G_1$ in Fig. 1, we see that there are two branches in which delete is immediately followed by read and these two branches record different traces of observed events, namely, write.read* and read.busy*. The other branch, which does not satisfy the pattern, records write.read.busy*. Thus, after the observation of write.read.read or read.busy, we know for sure that the pattern has been recognized. However, we cannot predict the recognition of the pattern ahead of time. This is because after recording the trajectory write.read, the system may execute busy and evolve into the state "file_1"; in this case, if we wait for read to occur afterwards, then this means that we are in the cycle formed by the states "file_0", "process_E", and "process_F", and the pattern has already occurred.* •

We now formally define the predictability of patterns. As previously said, this generalizes the predictability definition introduced in [10] by considering sequence patterns instead of single events.

**Definition 7** *An LTS $G$ is* $\Omega$-predictable, *whenever*

$$\begin{array}{l}\exists n \in \mathbb{N},\ \forall s \in \mathcal{L}(G) \cap \mathcal{L}_{F_\Omega}(\Omega) \cap \Sigma^*.\Sigma_o, \\ \exists t \in (\mathcal{L}(G) \cap \Sigma^*.\Sigma_o) \cup \{\epsilon\},\ t < s\ \wedge\ t \notin \mathcal{L}_{F_\Omega}(\Omega) \\ \qquad \textit{such that} \\ \forall u \in [\![P(t)]\!], \forall v \in \mathcal{L}(G)/u, \|P(v)\| \geq n \Rightarrow u.v \in \mathcal{L}_{F_\Omega}(\Omega)\end{array}$$

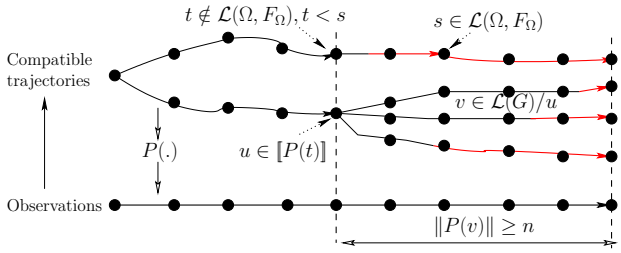



Figure 3: Intuitive explanation of predictability

◇

The definition means that for any trajectory $s$ recognized by the pattern, there exists a strict prefix $t$ not recognized by the pattern, such that any trajectory $u$ compatible with observation $P(t)$ will inevitably be extended into a trajectory $u.v$ recognized by the pattern. That is, upon observation of $P(t)$, one can already predict that the pattern, while not yet recognized, will inevitably be recognized; see Figure 3 for a conceptual representation.

**Example 2** *We consider a variation of Example 1. The new system is given by the LTS $G_2$ represented in Fig. 4. The pattern $\Omega$ and the partition of the event set into observable and unobservable events are the same as in Example 1 (see Fig. 2). The occurrence of the pattern $\Omega$ is predictable in $G_2$. In order to simplify the notation, we denote the events with their first letters in the following. Note that $\mathcal{L}(G_2) \cap \mathcal{L}_{F_\Omega}(\Omega) \cap \Sigma^*.\Sigma_o = w.r.c.d.r.(c.d.r)^* \cup c.w.r.d.r.(d.r)^*$.*

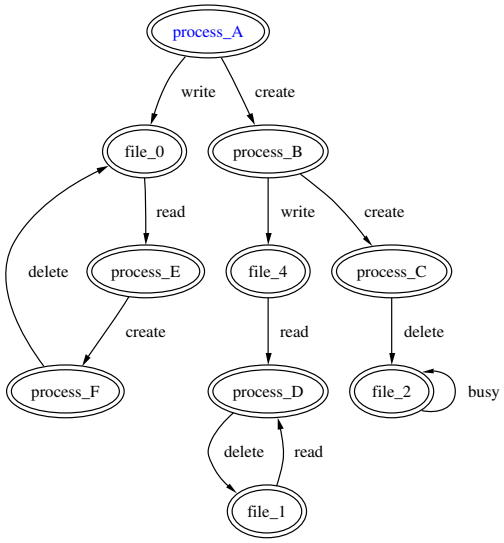



Figure 4: The LTS $G_2$ of Example 2

- *Consider any $s \in w.r.c.d.r.(c.d.r)^*$. Pick $t = w.r$. Then, $t < s$ and $t \notin \mathcal{L}_{F_\Omega}(\Omega)$. We have $[\![P(t)]\!] = \{w.r,\ c.w.r\}$. For $u = w.r$, and $v \in \mathcal{L}(G_2)/u$, $\|P(v)\| \geq 1$, we have $u.v \subseteq w.r.(c.d.r)^* \subseteq \mathcal{L}_{F_\Omega}(\Omega)$. For $u = c.w.r$, and $v \in \mathcal{L}(G_2)/u$, $\|P(v)\| \geq 1$, we obtain $u.v \in c.w.(d.r)^* \subseteq \mathcal{L}_{F_\Omega}(\Omega)$. Thus, there exists $t$ and $n$ (=1) satisfying the definition of $\Omega$-predictability for $s$.*
- *For any $s \in c.w.r.d.r.(d.r)^*$, it is sufficient to consider $t = c.w.r$ and $n = 1$ to satisfy the predictability condition.*

*Thus, $G_2$ and $\Omega$ satisfy the definition of predictability.* •

Let us now relate the notion of $\Omega$-predictability to the notion of $\Omega$-diagnosability of Section 3.

**Proposition 1** *Given a system $G$ and a pattern $\Omega$, if $G$ is $\Omega$-predictable, then $G$ is also $\Omega$-Diagnosable.*

**Proof** The proof of the proposition immediately follows from the respective definitions of these properties. (Further details on pattern diagnosability can be found in [14, 9].) ◇

The converse of this proposition is false. A counter-example is provided by Example 1.

## 5 Verification of Predictability

In this section, we present an off-line algorithm to verify the property of predictability. In [9], where the problem of predictability of single event occurrences is considered, the algorithm for the verification of predictability is based on the construction of a diagnoser automaton. In this work, we propose to adapt the polynomial-time verification of diagnosability based on *verifier automata* [12, 20] to the verification of predictability. In [14], the verifier was used to verify the diagnosability of a pattern in polynomial time in the number of states of the system.

In the case of diagnosability, the verifier identifies the existence, if any, of strings that violate the definition of this property. We adopt this underlying principle in building the algorithm for verification of predictability of patterns. By definition of predictability, the trajectories that violate predictability are the ones that: (i) are accepted by the pattern, (ii) end with observable events, and (iii) have the property that none of their prefixes are sufficient to predict the occurrence of the pattern. That is, for all the prefixes of such trajectories, there exists a trajectory that produces the very same observation as the prefix but whose continuation does not contain the pattern. Thus, if we find one or more trajectories violating predictability as was just described, then we can conclude that at least one occurrence of the pattern is not predictable in the system. It is possible to transform the search for these trajectories into a search for states in an LTS that carries information on acceptance, inevitability, and projections, for trajectories leading to them.

In the remainder of this section, we decompose the verification of predictability into five steps, each corresponding to a particular operation required for the verification. We start with system $G = (Q_{\mathrm{G}}, \Sigma, \rightarrow_{\mathrm{G}}, q^0_{\mathrm{G}})$ and pattern $\Omega = (Q_\Omega, \Sigma, \rightarrow_\Omega, q^0_\Omega)$ equipped with $F_\Omega$.

**Algorithm for Verification of Predictability:**

**Step 1** Construct the synchronous product $G_\Omega = G \times \Omega = (Q_{G_\Omega}, \Sigma, \rightarrow_{G_\Omega}, q^0_{G_\Omega})$ and the final state set $F = Q_{\mathrm{G}} \times F_\Omega$. By the properties of synchronous product, and using the fact that $\Omega$ is complete (thus $\mathcal{L}(\Omega) = \Sigma^*$), we get $\mathcal{L}(G_\Omega) = \mathcal{L}(G) \cap \mathcal{L}(\Omega) = \mathcal{L}(G)$ and $\mathcal{L}_F(G_\Omega) = \mathcal{L}(G) \cap \mathcal{L}_{F_\Omega}(\Omega)$. Thus, the accepted trajectories of $G_\Omega$ in $F$ are exactly the trajectories of $G$ accepted by $\Omega$ in $F_\Omega$. Moreover, note that $F$ is stable in $G_\Omega$ as both $Q_{\mathrm{G}}$ and $F_\Omega$ are stable by assumption.

**Step 2** Compute $Inev_{G_\Omega}(F)$ and consider the following partition of the state space $Q_{G_\Omega}$

$$Q_{G_\Omega} = N \cup I \cup F$$

where $I = Inev_{G_\Omega}(F) \setminus F$ is the set of states not belonging to $F$ but from which $F$ is unavoidable, and $N = Q_{G_\Omega} \setminus Inev_{G_\Omega}(F_{G_\Omega})$ is the set of states from which $F$ is avoidable.

The diagram of Fig. 5 shows how $G_\Omega$ evolves from one state to another according to the set of states it belongs to. This diagram illustrates the fact that even though the system can remain forever in $N$-states, if it reaches an $I$-state, then after a finite number of steps the system will eventually evolve into an $F$-state.

The construction of $G_{1\Omega}$ of Example 1 is illustrated in Fig. 6, with $N$-states unshaded, $F$-states dark-shaded, and $I$-states light-shaded with dashed-line borders.

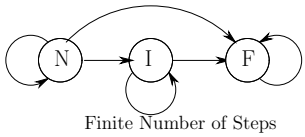



Figure 5: Possible transitions in the partition of $Q_{G_\Omega}$

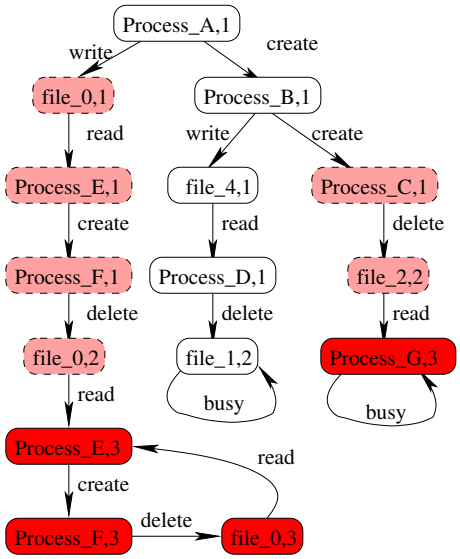



Figure 6: The LTS $G_{1\Omega}$

**Step 3** Construct the Unobservable-Closure (see Def. 3) $\text{Uc}(G_\Omega) = (Q_{G_\Omega}, \Sigma_o, \rightarrow_{\text{Uc}}, q^0_{G_\Omega})$ of $G_\Omega$ with final state set $F$ and preservation of the partition $Q_{G_\Omega} = N \cup I \cup F$. Using this partition, label states in $(Pre^{\exists}_{\text{Uc}(G_\Omega)}(F) \setminus F)$, corresponding to the set of states having an immediate successor in $F$ but not belonging to $F$. This set characterizes the maximal prefixes $t$ of trajectories of $G$ not recognized by $\Omega$ in $F_\Omega$, i.e., their observation is the last chance to predict acceptance of the pattern.

By the properties of Uc, we get $\mathcal{L}(\text{Uc}(G_\Omega)) = P(\mathcal{L}(G_\Omega))$, $\mathcal{L}_F(\text{Uc}(G_\Omega)) = P(\mathcal{L}_F(G_\Omega))$, and $\mathcal{L}_F(\text{Uc}(G_\Omega)) = P(\mathcal{L}_F(G_\Omega))$, which also means that Uc preserves information on the inevitability of the occurrence of the pattern, while abstracting out internal events.

Consider Example 1 again. The Unobservable-Closure of $G_{1\Omega}$ is shown in Fig. 7 where states in $(Pre^{\exists}_{\text{Uc}(G_\Omega)}(F) \setminus F)$ are those with thick borders.

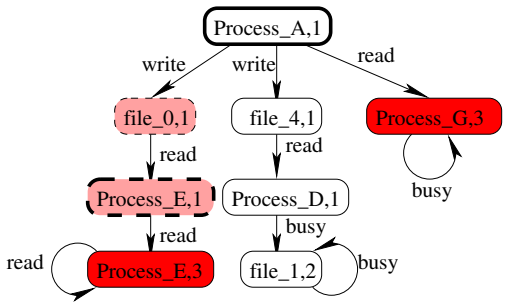



Figure 7: The LTS $\text{Uc}(G_{1\Omega})$

**Step 4** Construct the LTS $\Gamma = \text{Uc}(G_\Omega) \times \text{Uc}(G_\Omega) = (Q_{G_\Omega} \times Q_{G_\Omega}, \Sigma_o, \rightarrow_\Gamma, (q^0_{G_\Omega}, q^0_{G_\Omega}))$.

The role of $\Gamma$ is to synchronize trajectories of $G_\Omega$ with same trace $\mu$. By the definitions of Uc and synchronous product, if $\mu \in \mathcal{L}(G_\Omega)$ and $(q^0_{G_\Omega}, q^0_{G_\Omega}) \xrightarrow{\mu}_\Gamma (q, q')$, then there exists $s, s' \in [\![\mu]\!]$ s.t. $q^0_{G_\Omega} \xrightarrow{s} q$ and $q^0_{G_\Omega} \xrightarrow{s'} q'$ in $G_\Omega$.

**Step 5** Check the predictability condition given by Theorem 1 below. This completes the statement of the verification algorithm. ◇

**Theorem 1** *$G$ is $\Omega$-predictable iff $(Pre^{\exists}_{\mathrm{Uc}(G_\Omega)}(F) \setminus F) \times N$ is not reachable from $(q^0_{G_\Omega}, q^0_{G_\Omega})$ in $\Gamma$.*

A reachable state in $(Pre^{\exists}_{\mathrm{Uc}(G_\Omega)}(F) \setminus F) \times N$ characterizes two trajectories with same observation, one where $F_\Omega$ is avoidable, and the other one where it was the last chance to predict acceptance. Observe that the symmetry in $\Gamma$ allows us to simplify the condition, but in practice, reachable states in $N \times (Pre^{\exists}_{\mathrm{Uc}(G_\Omega)}(F) \cap \setminus F)$ should also be detected.

**Proof** ($\Rightarrow$) Suppose that $\Gamma$ has a reachable state $(q, q')$ in $(Pre^{\exists}_{\mathrm{Uc}(G_\Omega)}(F) \setminus F) \times N$. Note that we may have $q = q'$. On the one hand, there exists a trace $\mu \in \Sigma_o^*$, an observable event $\sigma_o \in \Sigma_o$, and states $q \in Pre^{\exists}_{\mathrm{Uc}(G_\Omega)}(F) \setminus F$, $q_1 \in F$ such that $q^0_{G_\Omega} \xrightarrow{\mu}_{\mathrm{Uc}} q \xrightarrow{\sigma_o}_{\mathrm{Uc}} q_1$. On the other hand, there exists a state $q' \in N$ such that $q^0_{G_\Omega} \xrightarrow{\mu}_{\mathrm{Uc}} q'$. By construction of $\mathrm{Uc}(G_\Omega)$ from $G_\Omega$, there exists $s', u' \in \Sigma^*$ with $P(s') = P(u') = \mu$ (at this point $s'$ may be identical to $u'$) and such that $q^0_{G_\Omega} \xrightarrow{s'}_\Gamma q \xrightarrow{\sigma_o}_\Gamma q_1$ and $q^0_{G_\Omega} \xrightarrow{u'}_\Gamma q'$. By taking $s = s'.\sigma_o$, for any $t \in \Sigma^*.\Sigma_o$ such that $t < s$, we have that $t \in \mathcal{L}(G)$ and $t \notin \mathcal{L}_{F_\Omega}(\Omega)$. But since $P(s') = P(u')$, there exists $u \leq u'$ such that $P(t) = P(u)$. Finally, $u \notin \mathcal{L}_{I \cup F}(G \times \Omega)$, otherwise this would imply that $q' \in I \cup F$ while $q' \in N$. This proves that $G$ is not $\Omega$-predictable.

($\Rightarrow$) Note first that $\Omega$-predictability is equivalent to $\forall s \in \mathcal{L}_{F_\Omega}(\Omega) \cap \mathcal{L}(G) \cap \Sigma^*.\Sigma_o$, $\exists t \in (\mathcal{L}(G) \cap \Sigma^*.\Sigma_o) \cup \{\epsilon\}$, $t < s \ \wedge \ t \notin \mathcal{L}_{F_\Omega}(\Omega)$ s.t. $[\![P(t)]\!] \subseteq \mathcal{L}_{Inev(Q_G \times F_\Omega)}(G \times \Omega)$.

Suppose now that $G$ is not $\Omega$-predictable. By the previous statement, $\exists s \in \mathcal{L}_{F_\Omega}(\Omega) \cap \mathcal{L}(G) \cap \Sigma^*.\Sigma_o$, such that $\forall t \in (\mathcal{L}(G) \cap \Sigma^*.\Sigma_o) \cup \{\epsilon\}$, $t < s \ \wedge \ t \notin \mathcal{L}_{F_\Omega}(\Omega)$ and $[\![P(t)]\!] \not\subseteq \mathcal{L}_{Inev(Q_G \times F_\Omega)}(G \times \Omega)$. Let $s = s'.s_{uo}.\sigma_o$ with $s' \in \Sigma^*.\Sigma_o$, $s_{uo} \in \Sigma_{uo}^*$, $\Sigma_o \in \Sigma_o$. By construction of $\mathrm{Uc}(G_\Omega)$, there exist states $q \in Pre^{\exists}_{\mathrm{Uc}(G_\Omega)}(F) \setminus F$, $q_1 \in F$ and $q' \in N$ such that $q^0_{G_\Omega} \xrightarrow{P(s')}_{\mathrm{Uc}} q \xrightarrow{\sigma_o}_{\mathrm{Uc}} q_1$ and $q^0_{G_\Omega} \xrightarrow{P(s')}_{\mathrm{Uc}} q'$. Thus, in $\Gamma$, $(q^0_{G_\Omega}, q^0_{G_\Omega}) \xrightarrow{P(s')}_\Gamma (q, q')$ with $(q, q') \in (Pre^{\exists}_{\mathrm{Uc}(G_\Omega)}(F) \setminus F) \times N$. ◇

Figures 8 and 9 depict $\Gamma_1$ and $\Gamma_2$ for $G_1$ and $G_2$, respectively, with the pattern $\Omega$.

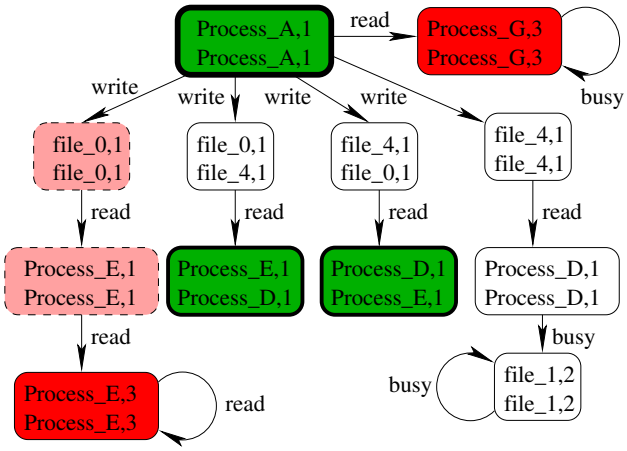



Figure 8: The LTS $\Gamma_1$ derived from $G_{1_\Omega}$

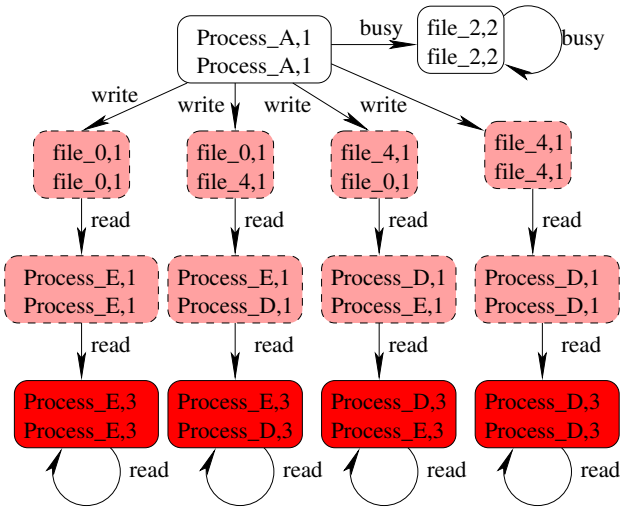



Figure 9: The LTS $\Gamma_2$ derived from $G_{2_\Omega}$

The three states with thick borders in $\Gamma_1$ are states in $(Pre^{\exists}_{\mathrm{UC}(G_\Omega)}(F) \setminus F) \times N$ and prove that $G_1$ is not $\Omega$-predictable for two reasons: (i) String *create.create.delete.read* with observation *read* already leads to acceptance, but acceptance is not initially predictable because of trajectories in *create.write.read.delete.busy*$*$; (ii) Even if *create.create.delete.read* was suppressed in $G_1$, there still exists the trajectory *write.read.create.delete.read* that is recognized by $\Omega$. Its maximal prefix ending in an observation and not recognized by $\Omega$ is *write.read*. The observation *write.read* also corresponds to the trajectory *create.write.read* for which the acceptance is never achieved.

Regarding $G_2$, as was already mentioned earlier, it is $\Omega$-predictable. Indeed, there are no reachable states in $(Pre^{\exists}_{\mathrm{UC}(G_\Omega)}(F) \setminus F) \times N$ in $\Gamma_2$ shown in Fig. 9.

Finally, note that both $G_1$ and $G_2$ are $\Omega$-diagnosable, as in either $\Gamma_1$ and $\Gamma_2$, there are no cycles of states of the form $(I \cup N) \times F$ (see [14] for further details on this verification of diagnosability).

We summarize the above algorithm to verify the predictability of the occurrences of a pattern $\Omega$ in a system $G$ and denote it by *Verif_Pred* hereafter:

*Verif_Pred*($G$,$\Omega$)
- (i) Computation of $G_\Omega = G \times \Omega$
- (ii) Computation of the state sets $F$, $I$ and $N$ in $G_\Omega$
- (iii) Computation of $\mathrm{UC}(G_\Omega)$ and of the state set $(Pre^{\exists}_{\mathrm{UC}(G_\Omega)}(F) \setminus F)$
- (iv) Computation of $\Gamma = \mathrm{UC}(G_\Omega) \times \mathrm{UC}(G_\Omega)$
- (v) Test of the emptiness of $(Pre^{\exists}_{\mathrm{UC}(G_\Omega)}(F) \setminus F) \times N$.

**Proposition 2** *The complexity of the verification of predictability using Verif_Pred($G$,$\Omega$) is quadratic in the product of the sizes of $G$ and $\Omega$.*

The proof of this result is based on the following analysis of the steps within *Verif_Pred*($G$,$\Omega$) . The first step is the construction of $G_\Omega = G \times \Omega$, which is linear in $\|G\| \times \|\Omega\|$ where $\|G\|$ and $\|\Omega\|$ are the sizes of $G$ and $\Omega$ in terms of states and transitions. The next step is the computation of $Inev_{G_\Omega}(F)$ to partition $Q_{G_\Omega}$ into $N \cup I \cup F$. This is linear in $\|G_\Omega\|$. The third step consists of constructing $\mathrm{UC}(G_\Omega)$ and labelling of states by $(Pre^{\exists}_{\mathrm{UC}(G_\Omega)}(F) \setminus F)$. It is linear in $\|G_\Omega\|$. The construction of $\Gamma = \mathrm{UC}(G_\Omega) \times \mathrm{UC}(G_\Omega)$ in the fourth step is quadratic in the size of $\mathrm{UC}(G_\Omega)$. Checking that no reachable state is in $(Pre^{\exists}_{\mathrm{UC}(G_\Omega)}(F) \setminus F) \times N$ (Step 5) can be done during this construction. Thus,

the complexity of the entire verification of predictability is quadratic in the product of the sizes of $G$ and $\Omega$.

## 6 The P_diagnoser

In this section, we explain how to construct from $G$ and $\Omega$ a special kind of diagnoser, called P_diagnoser (for Prediction Diagnoser), that can be used on-line to predict the recognition of a pattern. We emphasize some properties of this P_diagnoser when the occurrence of the pattern $\Omega$ is predictable in $G$.

Consider $G_\Omega$ as defined in Section 5 (Step 1) and build $Det(G_\Omega) = (\mathcal{X}, \Sigma_o, \rightarrow_d, X^0)$ as defined in Definition 4. Remember that this can be done by reusing $\textsc{Uc}(G_\Omega)$ and applying a subset construction. Then, we have that $\mathcal{L}(Det(G_\Omega)) = P(\mathcal{L}(G_\Omega)) = P(\mathcal{L}(G))$. Thus, for any observation $\mu \in P(\mathcal{L}(G))$, $\Delta_{Det(G_\Omega)}(X^0, \mu) = \{\Delta_{G_\Omega}(q^0_{G_\Omega}, [\![\mu]\!])\}$.

The P_diagnoser for predictability, denoted by $\texttt{P\_Diag}_\Omega$, is defined to be $Det(G_\Omega)$ together with a decision function on its state space that is defined as follows. For any observation $\mu$ in $P(\mathcal{L}(G))$, let the decision function of $\texttt{P\_Diag}_\Omega(\mu)$ be:

$$\begin{cases} Yes, & \text{if } \Delta_{Det(G_\Omega)}(X^0, \mu) \subseteq F \\ Pred, & \text{if } \Delta_{Det(G_\Omega)}(X^0, \mu) \subseteq I \\ Pred\_F, & \text{if } \Delta_{Det(G_\Omega)}(X^0, \mu) \subseteq (I \cup F) \\ & \quad \wedge\ \Delta_{Det(G_\Omega)}(X^0, \mu) \cap I \neq \emptyset \\ & \quad \wedge\ \Delta_{Det(G_\Omega)}(X^0, \mu) \cap F \neq \emptyset \\ No, & \text{if } \Delta_{Det(G_\Omega)}(X^0, \mu) \subseteq N \\ Pred\_N & \text{if } \Delta_{Det(G_\Omega)}(X^0, \mu) \subseteq (I \cup N) \\ & \quad \wedge\ \Delta_{Det(G_\Omega)}(X^0, \mu) \cap I \neq \emptyset \\ & \quad \wedge\ \Delta_{Det(G_\Omega)}(X^0, \mu) \cap N \neq \emptyset \\ ?, & \text{otherwise.} \end{cases} \tag{1}$$

In order to better understand the decisions of the P_diagnoser, we consider a language point of view. Using the partition of $Q_{G_\Omega} = N \cup I \cup F$, we can partition $\mathcal{L}(G) = \mathcal{L}(G_\Omega)$ into three different languages

$$\mathcal{L}(G) = \mathcal{L}_N(G_\Omega) \cup \mathcal{L}_I(G_\Omega) \cup \mathcal{L}_F(G_\Omega)$$

Based on this partition, we then have the following results that are derived directly from the definition on the P_diagnoser:

- $\texttt{P\_Diag}_\Omega(\mu) = Yes$ if $[\![\mu]\!] \subseteq \mathcal{L}_F(G_\Omega)$
- $\texttt{P\_Diag}_\Omega(\mu) = Pred$ if $[\![\mu]\!] \subseteq \mathcal{L}_I(G_\Omega)$
- $\texttt{P\_Diag}_\Omega(\mu) = Pred\_F$ if
  - $[\![\mu]\!] \subseteq \mathcal{L}_I(G_\Omega) \cup \mathcal{L}_F(G_\Omega)$, and
  - $[\![\mu]\!] \cap \mathcal{L}_I(G_\Omega) \neq \emptyset$, and
  - $[\![\mu]\!] \cap \mathcal{L}_F(G_\Omega) \neq \emptyset$
- $\texttt{P\_Diag}_\Omega(\mu) = No$ if $[\![\mu]\!] \subseteq \mathcal{L}_N(G_\Omega)$

- $\texttt{P\_Diag}_\Omega(\mu) = Pred\_N$ if
  - $[\![\mu]\!] \subseteq \mathcal{L}_N(G_\Omega) \cup \mathcal{L}_I(G_\Omega)$, and
  - $[\![\mu]\!] \cap \mathcal{L}_N(G_\Omega) \neq \emptyset$, and
  - $[\![\mu]\!] \cap \mathcal{L}_I(G_\Omega) \neq \emptyset$

$Yes$ means that all the trajectories in $[\![\mu]\!]$ lie in $\mathcal{L}_{F_\Omega}(\Omega)$. That is, the observation $\mu$ only corresponds to trajectories recognized by the pattern. $Pred$ means trajectories compatible with $\mu$ are not recognized by the pattern so far, but all their extensions will inevitably do so after a bounded number of observations. $No$ simply means that all trajectories compatible with $\mu$ can still be extended without being recognized by the pattern. $Pred\_F$ means that the recognition of the pattern is inevitable but some trajectories compatible with the observation are already recognized by the pattern. $Pred\_N$ means that the observed trace corresponds to some trajectories where recognition of the pattern is inevitable, but others where this is false.

Once again, we use $G_1$ and $G_2$ of Examples 1 and 2 to illustrate these results. The P_diagnosers (as well as their verdicts) obtained by determinization of $G_{1\Omega}$ and $G_{2\Omega}$ for $G_1$ and $G_2$, respectively, with the pattern $\Omega$ are given in Fig. 10.

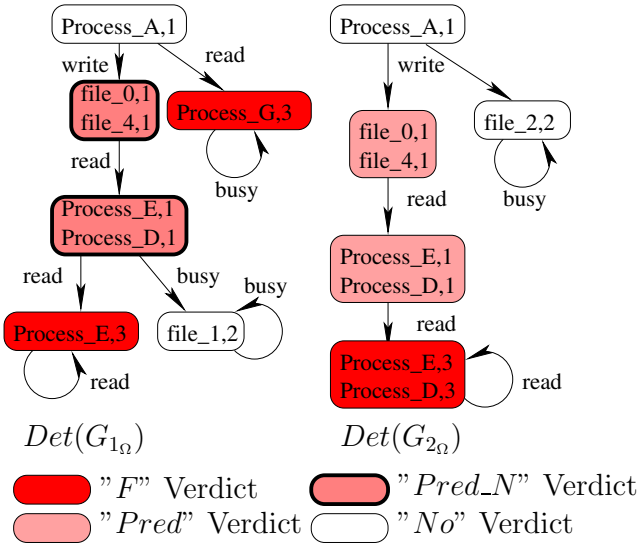



Figure 10: The P_diagnosers

We have the following results about P_diagnosers.

**Proposition 3** *If $G$ is $\Omega(n)$-predictable, then:*

**(i)** $\texttt{P\_Diag}_\Omega(\mu) = Pred \;\Rightarrow\; (\forall \mu' \in P(\mathcal{L}(G))/\mu,\ \|\mu'\| \geq n \Rightarrow \texttt{P\_Diag}_\Omega(\mu\mu') = Yes)$

**(ii)** $\texttt{P\_Diag}_\Omega(\mu) = Yes \Rightarrow \exists \mu' < \mu$ *such that* $\texttt{P\_Diag}_\Omega(\mu') = Pred$ *and* $\|\mu\| - \|\mu'\| \leq n$

**(iii)** $\forall \mu \in P(\mathcal{L}(G)),\ \texttt{P\_Diag}_\Omega(\mu) \neq ?$ ◇

Part (i) of Proposition 3 simply says that when the P_diagnoser issues the decision $Pred$, after at most $n$ observations, the P_diagnoser will issue the decision $Yes$. Part (ii) emphasizes the fact that if

the P_diagnoser issues the decision $Yes$, then in the past the P_diagnoser already issued the decision $Pred$. Part (iii) shows that the decision "?" cannot be issued when the system is $\Omega$-predictable. Indeed, if this decision were to be issued, it would mean that there is a trajectory compatible with $\mu$ that is recognized by the pattern without having being detected, which contradicts the fact that the system is predictable.

Figure 11 explains the possible evolutions of the decisions issued by the P_Diagnoser whenever the system is predictable.

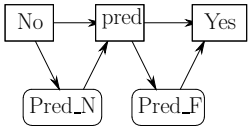



Figure 11: Possible evolutions of the decisions of the P_diagnoser

## 7 Comments on Predictability and P_Diagnoser

**More expressive models.** In this paper, we have considered so far, system and pattern modeled by finite LTS and thus by regular languages. This restriction can be easily raised. In the definition of predictability (Def.7), it would be possible to consider non regular languages to model the system and the pattern. One can indeed consider a language-based pattern $K \subseteq \Sigma^*$ as far as this language is extension-closed, i.e. such that $K\Sigma^* = K$ and any prefix-closed language $L$ to model the system. With these notations, the $K$-predictability of $L$ would be given by

$$\begin{array}{l}\exists n \in \mathbb{N}, \forall s \in L \cap K \cap \Sigma^*.\Sigma_o, \\ \quad \exists t \in (L \cap \Sigma^*.\Sigma_o) \cup \{\epsilon\},\ t < s\ \wedge\ t \notin K, \text{ such that} \\ \forall u \in [\![P(t)]\!], \forall v \in L/u, \|P(v)\| \geq n \Rightarrow u.v \in K\end{array}$$

With this simple extension, more expressive pattern (as well as systems) can be considered. However, even though interesting from a theoretical point of view, the verification of predictability requires fix-point computations (for inevitability) which may not terminate, and the construction of the corresponding P_diagnoser requires determinization which may not be feasible (See Sections 5 & 6). For example if we consider extended automata with variables to model the system and the pattern, then this may force to rely on over-approximations analysis [4] for inevitability and restrictions to deterministic models [15].

**Comparison with [13].** The results we presented in this paper are very close the work of Jiang and Kumar [13] who introduced the notion of *inevitability*. Indeed, in [13], the authors are interested in diagnosing inevitability, thus in detecting that a pattern will be inevitably satisfied, within a bounded number of observations *after* after this inevitability[2], while prediction means detecting inevitability strictly before its occurrence. It should also be noted that, despite the fact that the diagnoser for inevitability may give earlier verdicts than the diagnoser for satisfaction, the diagnosability of inevitability is strictly equivalent to diagnosability of satisfaction. In contrast, we have shown that predictability implies diagnosability, but not the converse. Morevoer, one consequence of their definition of diagnosability is that the on-line diagnoser is not able to infer whether the pattern has been actually recognized or not. Indeed, compared to the P_diagnoser, the verdicts $Pred$, $Pred\_F$ and $F$ are merged in a single verdict.

[2]This work considers properties specified in Linear Temporal Logic (LTL) which are more expressive as patterns. However, their notion of pre-diagnosability required for diagnosability means that the property reduces to a stable property on the system (can be checked on a finite trajectory).

## 8 Conclusion

We presented new results on predictability in DES. We defined the notion of predictability of occurrences of patterns modeled as sequences of events in an LTS. We presented an off-line algorithm to verify the predictability of a pattern and an on-line algorithm to analyze the occurrences of the pattern during the operation of the system. We showed that the off-line algorithm is polynomial in the number of states of the system. While this has not been done yet, our predictability verification could be implemented within the software environment DESUMA [19].

In this study, we restricted attention to systems and patterns that can be expressed by regular languages. The definition of predictability of patterns can easilly be extended to include systems and patterns expressed by non-regular languages (See Section 7). However, the development of verification and on-line prediction algorithms for systems or patterns that are not modeled by regular languages remains an open problem.

## 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.-R. 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] P. Cousot and R. Cousot. Systematic design of program analysis frameworks. In *6th ACM Symposium on Principles of Programming Languages, POPL'79*, San Antonio, January 1979.

[5] 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.

[6] P. Declerck. Predictability and control synthesis in time deviant graphs. In *Workshop on Discrete-Event Systems*, Cagliari, Italy, August 1998.

[7] 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.

[8] P. H. Feiler, B. Lewis, and S. Vestal. Improving predictability in embedded real-time systems, December 2000.

[9] 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, December 2006.

[10] S. Genc and S. Lafortune. Predictability in discrete-event systems under partial observation. In *IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*, Beijing, China, August 2006.

[11] 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.

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

[13] S. Jiang and R. Kumar. Failure diagnosis of discrete event systems with linear-time temporal logic fault specifications. *IEEE Transactions on Automatic Control*, 49(6):934–945, 2004.

[14] T. Jéron, H. Marchand, S. Pinchinat, and M-O. Cordier. Supervision patterns in discrete event systems diagnosis. In *Workshop on Discrete Event Systems, WODES'06*, Ann-Arbor (MI, USA), July 2006.

[15] T. Jéron, H. Marchand, and V. Rusu. Symbolic determinisation of extended automata. In *4th IFIP International Conference on Theoretical Computer Science*, IFIP book series, Stantiago, Chile, August 2006. Springer Science and Business Media.

[16] S. T. King and P. M. Chen. Backtracking intrusions. *ACM Trans. Comput. Syst.*, 23(1):51–76, February 2005.

[17] S. Lafortune, D. Teneketzis, M. Sampath, R. Sengupta, and K. Sinnamohideen. Failure diagnosis of dynamic systems: An approach based on discrete event systems. In *Proc. 2001 American Control Conference*, pages 2058–2071, June 2001.

[18] M. Musolesi and C. Mascolo. Evaluating context information predictability for autonomic communication. In *American Control Conference*, Minneapolis, Minnesota USA, June 2006.

[19] 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.

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