Discriminability Analysis of Supervision Patterns by Net Unfoldings

Houssam-Eddine Gougam · Audine Subias · Yannick Pencolé†

† CNRS; LAAS; 7 avenue du Colonel Roche, F-31400 Toulouse, France
Univ de Toulouse, INSA, LAAS, F-31400 Toulouse, France

Abstract: In this paper, we are interested in the discriminability of supervision patterns, in discrete event systems (DES). Discriminability — as opposed to diagnosability — is the possibility to detect the exclusive occurrence of a particular behavior of interest — called the supervision pattern. To this end, we propose to adapt the classical twin-plant approach to Petri nets unfolding. The usage of unfoldings permits us to avoid the combinatorial explosion associated with marking graphs. The method can also be used to solve the classical problem of discrete event systems’ diagnosability.

Keywords: discriminability, diagnosability, supervision patterns, labeled Petri nets, Petri nets unfolding.

1. INTRODUCTION

In the context of discrete-event systems (DES), the supervision task consists in analyzing the sequence of observed events and determining whether an abnormal/faulty situation has occurred before deciding what kind of actions to perform in order to recover the optimal performance of the system. By the intrinsic nature of DES, an abnormal situation is characterized by a partial order of observable/non-observable events called an event pattern or a supervision pattern as introduced in Jéron et al. (2006).

In this paper, we address the problem of analysing the discriminability of a set of supervision patterns. Two supervision patterns \( R_1 \) and \( R_2 \) are discriminable if it is always possible to assert from the observed events that if \( R_1 \) (resp. \( R_2 \)) has occurred then \( R_2 \) (resp. \( R_1 \)) has not occurred and will not occur.

The notion of pattern discriminability is obviously related to the notion of pattern diagnosability and their respective analyses (Jiang et al. (2003), Jéron et al. (2006), Yoo and Garcia (2008), Gougam et al. (2013)), however the point of view is different. The result of a diagnosability analysis states whether any considered pattern is always detectable or not. If such a property holds then it is possible to design a diagnoser that will always be able to determine which patterns have occurred in a finite amount of time. However there is a pitfall: underlying real-world systems are not and cannot be diagnosable as the diagnosability property is very restrictive (Pencolé (2005)) and requires an observability level that cannot be implemented as a set of sensors on the underlying system (physical and/or cost constraints). The classical diagnosability analyses will then conclude such a real-world system is not diagnosable, so what is next?

To cope with this problem, we propose to develop a more constructive analysis by checking the discriminability of the occurrence of a set of patterns with respect to another disjoint set of patterns. The analysis is constructive in the sense that any partial discriminability result has an outcome that can impact the design of a diagnoser for the analysed system even if the underlying system is not globally diagnosable. And finally, once the proposed discriminability analysis is completed, checking the classical diagnosability analysis is straightforward and for free.

In our proposal, the development of the discriminability analysis relies on two original choices. The first one is the use of Petri net — which were shown to be more appropriate to solve diagnosis problems Lai et al. (2008) — to model patterns as well as the system (Basile et al. (2009), Dotoli et al. (2009)) so that we benefit of the natural way to represent event concurrency. The second one is the use of net unfoldings (McMillan (1995), Espanza et al. (2002), Benveniste et al. (2003)) in order to benefit of a representation as a partial order of events. Unfoldings avoid the explicit enumeration of event sequences that is performed by techniques based on marking graphs like in Cabasino et al. (2012). Our proposed approach allows us to have some interesting properties. First, genericness, meaning that we do not impose particular patterns, we rather define a generic framework for supervision pattern, and every pattern falling in this framework can be used for supervision. Second, the supervision patterns are compact, i.e. only relevant events are included leading to more concise patterns. Finally, reusability, which is a direct consequence of the compactness, the exclusive use of relevant events yields supervision patterns which are independent from the system.

The paper is organized as follows. Section 2 introduces the problem, Section 3 presents Petri nets and their use to model the system and the patterns. The proposed analysis method is detailed in Sections 4 and 5. An example of
our method is given in Section 7 and give some perspectives.

2. DISCRIMINABILITY OF SUPERVISION PATTERNS

Analyzing the discriminability of a system in the DES context means characterizing the performances of the diagnosis algorithm. A diagnosis algorithm, given an observable sequence produced by the system, is responsible of returning its state, i.e. is the system in a normal state or a faulty state, and in the latter case, which — faulty — behavior yielded this state. If such algorithm exists, and can determine with certainty after a bounded number of observations that only a particular combination of behaviors occurred, this combination is said to be discriminable. On the other hand, if the algorithm can detect the occurrence of a behavior with no information about the possible occurrence of other faulty behaviors, the former is said to be detectable.

In this paper, we consider a system modeled by a language, faulty behaviors are recognized by supervision patterns which are also languages with specific properties. More formally, the problem of supervision pattern discriminability in discrete event systems is then defined in the following context.

Let $\Sigma = \{a, b, \ldots\}$ be a finite set called an alphabet. The Kleene closure of $\Sigma$ denoted $\Sigma^*$ is the set of finite sequences — or words — over $\Sigma$ including the empty sequence, denoted in the following by $\lambda$. $\Sigma^+ = \Sigma^* \setminus \{\lambda\}$ is the set of non-empty finite sequences. A subset $\Sigma \subseteq \Sigma^*$ is called a language over $\Sigma$. The continuation $z$ of a sequence $w$ in $\Sigma$ is a sequence such that $wz \in \Sigma$. So, the set of $w$’s continuations in $\Sigma$ is defined as $\Sigma/w = \{z \in \Sigma^+ : wz \in \Sigma\}$.

The classical projection of a sequence $w \in \Sigma^*$ on a subset $\Sigma_p$ of $\Sigma$ is denoted $P_{\Sigma_p}(w)$. Finally, $\|w\|$ denotes the length of the sequence $w$. The behavior of the system is modeled by the prefix-closed language $\Sigma$ over an alphabet $\Sigma$ representing the set of events generated by the system. Some of these events are observable, while others are not. The observable events are represented by the set $\Sigma_o$ and the set of non-observable events is called $\Sigma_u$. So, $\Sigma = \Sigma_o \cup \Sigma_u$. We make the assumption that the system is $\Sigma_o$-alive, meaning: $\forall w \in \Sigma, \exists z \in \Sigma/w : P_{\Sigma_o}(z) \neq \lambda$; this hypothesis ensures that the system produces observations with some regularity. A supervision pattern $\mathcal{R}$ is a language over $\Sigma_R \subseteq \Sigma$ that recognizes $\mathcal{R}$-faulty sequences.

Definition 1. Let $\mathcal{R} = \{R_1, R_2, \ldots, R_n\}$ be a set of supervision patterns. A sequence $w \in \Sigma^*$ is said to be $\mathcal{R}$-faulty if: $P_{\Sigma_{R_i}}(w) \in R_i$. A sequence $w \in \Sigma^*$ is said to be $\mathcal{R}$-faulty if: $\forall R \in \mathcal{R}, w \in R$-faulty. A sequence $w \in \Sigma^*$ is said to be $\mathcal{R}$-faulty if: $\exists R \in \mathcal{R}$, $w$ is $R$-faulty.

Definition 2. Let $\mathcal{R} = \{R_1, R_2, \ldots, R_n\}$ be a set of supervision patterns. Let $\mathcal{R}^* \subseteq 2^\mathcal{R}$ be a subset of $\mathcal{R}$. A sequence $w \in \Sigma^*$ is said to be exclusively $\mathcal{R}^*$-faulty in $\mathcal{R}$ if:

\[
\forall R \in \mathcal{R}^* : w \text{ is } R \text{-faulty} \\
\forall R \notin \mathcal{R}^* : w \text{ is not } R \text{-faulty}
\]

We will simply use “exclusively $\mathcal{R}^*$-faulty” rather than “exclusively $\mathcal{R}^*$-faulty in $\mathcal{R}$” when there is no ambiguity.

We can now define the discriminability of a set of supervision patterns.

Definition 3. Given a system $\Sigma$ and a set of supervision patterns $\mathcal{R} = \{R_1, R_2, \ldots, R_n\}$. A subset $\mathcal{R}^*$ of $\mathcal{R}$ is said to be discriminable if:

$\exists n \in \mathbb{N}, \forall w \text{ an exclusively } \mathcal{R}^* \text{-faulty word of } \Sigma, \forall z \in \Sigma/w : \|P_{\Sigma_o}(z)\| \geq n \Rightarrow D$

where the discriminability condition $D$ is:

$\forall w^o \in P_{\Sigma_{R_i}}^{-1}(P_{\Sigma_o}(wz)) : w^o \text{ is exclusively } \mathcal{R}^*$-faulty

This property insures that one can detect with certainty that only a particular combination of patterns — namely $\mathcal{R}^*$ — occurred.

Similarly, we define the $\mathcal{R}$-diagnosability of a system.

Definition 4. Let $\mathcal{R} = \{R_1, R_2, \ldots, R_n\}$ be a set of supervision patterns. A language $\Sigma$ is $\mathcal{R}$-diagnosable if:

$\forall R \in \mathcal{R}(\exists n \in \mathbb{N}), \forall w \text{ an } R \text{-faulty word of } \Sigma, \forall z \in \Sigma/w : \|P_{\Sigma_o}(z)\| \geq n \Rightarrow D$

where the diagnosability condition $D$ is:

$\forall w^o \in P_{\Sigma_{R_i}}^{-1}(P_{\Sigma_o}(wz)) : w^o$ is $R$-faulty

More intuitively, a language $\Sigma$ is $\mathcal{R}$-diagnosable if it does not contain two arbitrary long sequences, the first $R$-faulty, the second non $R$-faulty, which have the same observable behavior.

3. ON MODELING ASPECTS

To tackle the problem of discriminability, we propose to use labeled Petri nets to model the system and the patterns.

3.1 Labeled Petri nets

Definition 5. A labeled Petri net is a tuple $(P, T, A, \ell, L, \Sigma)$ where:

- $P$: a set of places;
- $T$: a set of transitions with $P \cap T = \emptyset$;
- $A \subseteq (P \times T) \cup (T \times P)$: a binary relation representing arcs between nodes;
- $\ell : P \cup T \rightarrow L \cup \Sigma \cup \{\lambda\}$: a labeling function where $L$ is the set of place labels, $\Sigma$ is the set of transition labels and $\lambda$ denotes the empty sequence.
- $\ell$ is naturally extended to markings and sequences of nodes. Let $M$ be a marking: $\ell(M) = \{\ell(p) : p \in M\}$. For $s = s_1s_2 \ldots \in (P \cup T)^*$, $\ell(s) = \ell(s_1)\ell(s_2) \ldots$.

A marking $M$ is a map from $P$ to $\mathbb{N}$ which maps any place $p$ to the number of tokens $M(p)$ contained in it. For the sake of simplicity, a marking may sometimes be denoted as a multiset. For instance, let $P = \{p_1, p_2, p_3\}$, the marking $M$ such that $M(p_1) = 2, M(p_2) = 0$ and $M(p_3) = 1$ can be represented as $M = \{p_1, p_1, p_3\}$. A marked and labeled Petri net is a tuple $\Theta = (P, T, A, \ell, L, \Sigma, M_0)$ where $(P, T, A, \ell, L, \Sigma)$ is a labeled Petri net and $M_0$ an initial marking. The current state of a Petri net is defined by its current marking. The set $\bullet = \{p \in P : (p, t) \in A\}$ is the preset of $t$ and $\bullet = \{p \in P : (t, p) \in A\}$ is its postset (the preset $\bullet p$ and postset $p\bullet$ of a place $p$ are similarly defined). The transition $t$ is firable from a given marking $M$ iff: $\forall p \in \bullet : M(p) > 0$. Firing $t$ leads to a new marking $M'$ such that $M' = (M \setminus \bullet t) \cup \bullet$ and which is denoted by $M \xrightarrow{t} M'$. A marking $M$ is reachable if there exists a firing.
sequence \( s = t_0 t_1 \ldots t_n \) such that \( M_0 \xrightarrow{t_0} M_1 \xrightarrow{t_1} \ldots \xrightarrow{t_n} M \), we can also write \( M_0 \xrightarrow{\sim} M \). Given a marked and labeled Petri net \( \Theta \) and a set of final markings \( Q \), the G-type language (see Peterson (1977)) generated by \( \Theta \) is given by:

\[
L(\Theta) = \{ \ell(s) : s \in T^* \land M_0 \xrightarrow{\sim} M \land \exists M' \in Q, M' \subseteq M \}.
\]

The system is modeled by a Petri net \( \Theta \). As the language of the system is prefix-closed, it can be represented as the G-type language of a bounded Petri net \((\exists n \in \mathbb{N}, \forall p \in P : M(p) \leq n \})\) associated with the set of final markings \( Q = \{ \emptyset \} \) (any reachable marking being a superset of \( \emptyset \)).

3.2 Supervision Patterns

The classical framework of DES diagnosis (Sampath et al. (1995)) relies on detecting the simple occurrence of particular — faulty — events. The supervision patterns framework, as proposed in Jiang et al. (2003) and in Jéron et al. (2006), extend this analysis to more complex models, enabling us to express more interesting behaviors, possibly involving several events. We propose to represent a supervision pattern as a labeled Petri net.

Definition 6. A supervision pattern is a marked labeled Petri net \((P, T, A, \ell, L, \Sigma, M_0)\) such that:

1. (labeling) \( L = \{N, F\} \);
2. (initialization) \( \forall p \in P : M_0(p) > 0 \Rightarrow \ell(p) = N \);
3. (exclusivity) \( \forall p_1, p_2 \in P : \ell(p_1) = N \land \ell(p_2) = F \Rightarrow M(p_1) \times M(p_2) = 0 \);
4. (stability) for each reachable marking \( M \), \( \exists p \in P, M(p) > 0 \land \ell(p) = F \) implies that for each successor marking \( M' : \exists p' \in P, M'(p') > 0 \land \ell(p') = F \).
5. (completeness) for each reachable marking \( M \), \( \forall e \in \Sigma, \exists t \in T : \ell(t) = e \land \forall p \in \bullet t, M(p) > 0 \);

The first condition ensures that every place is either labeled by \( N \) or \( F \). Condition 2 states that in the initial state, each marked place is labeled by \( N \). Condition 3 states that a place labeled \( N \) and another one labeled \( F \) can not be marked at the same time in a reachable marking, meaning that the pattern can not be recognized and not recognized at the same time. Condition 4 ensures that once the pattern is recognized — the place labeled \( F \) is marked — it can not return to a non faulty marking. Condition 5 is the completeness condition: from any reachable marking, the supervision pattern can fire a transition by any event in its alphabet, this condition ensures compact independent pattern (the pattern does not include other events than those relevant to faulty behavior).

Any supervision patterns is associated to the set of final markings \( Q \) that contain only one marking \( Q = \{ \{p \in P : \ell(p) = F \}\} \).

Example: event concurrency Let \( E = \{e_1, e_2, \ldots \} \) be a set of events. The pattern describing the concurrent occurrence of these events is the labeled Petri net depicted in Figure 1, with the final marking \( Q = \{ \{p_0\} \} \). Transition \( t_0 \), as it is labeled with \( \lambda \), is fireable as soon as every event of \( E \) has occurred.

![Figure 1. A set of concurrent events.](image-url)
The completeness condition imposed on the supervision

Proposition 8.

Let the system. For clarity we will use a different symbol for
by a slight modification of the product — more precisely,
be taken into account within the system. This is realized
model all the possible combinations of their evolution —
context — the completeness condition states that in any
patterns ensures that this situation can not happen in our
arcs.

This product can be constructed simply by taking the union of
the two nets, removing transitions with labels in \( \Sigma_1 \cap \Sigma_2 \), and
for each \((t_1, t_2) \in T_1 \times T_2 \) where \( \ell(t_1) = \ell(t_2) \neq \lambda \), add a
transition \( t_1 \parallel t_2 \) to the product inheriting their label and arcs.

The product imposes the simultaneous evolution of the two
Petri nets, whenever a shared event occurs. In a general
case, this can lead to a deadlock — the case where
the first petri net needs to fire a synchronized transition
while the second one did not reach the appropriate response.
The completeness condition imposed on the supervision
patterns ensures that this situation can not happen in our
context — the completeness condition states that in any
marking, the pattern can fire a transition with any label.
So, applying this product on two supervision patterns can
model all the possible combinations of their evolution —
only one of them evolves, the two evolve in the same time,
etc.

**Final Markings: Product of Patterns**

Let \( Q_i \) be the set of final markings of \( \Omega_i, i = 1, 2 \). We define the set of
final markings of the product \( \Omega = \Omega_1 \parallel \Omega_2, Q = Q_1 \cup Q_2 \).

**Proposition 8.** \( \mathcal{L}(\Omega) = \left\{ w \in (\Sigma_1 \cup \Sigma_2)^* : p_{\Sigma_1}(w) \in \mathcal{L}(\Omega_1) \lor p_{\Sigma_2}(w) \in \mathcal{L}(\Omega_2) \right\} \).

**Proposition 9.** \( w \in \mathcal{L}(\Omega) \iff \exists w_1 \in \mathcal{L}(\Omega_1), p_{\Sigma_1 \cap \Sigma_2}(w) = p_{\Sigma_1 \cap \Sigma_2}(w_1) \lor \exists w_2 \in \mathcal{L}(\Omega_2), p_{\Sigma_1 \cap \Sigma_2}(w) = p_{\Sigma_1 \cap \Sigma_2}(w_2) \).

**Final Markings: Product of the System and a Pattern**

The recognition of the supervision pattern must be
taken into account within the system. This is realized by
a slight modification of the product — more precisely,
the final markings — which is applied to the pattern and
the system. For clarity we will use a different symbol for
this product \( \ast \).

Let \( Q_i \) be the set of final markings of \( \Theta_i, i = 1, 2 \). We define the set of
final markings of the product \( \Theta = \Theta_1 \circ \Theta_2, Q = Q_1 \otimes Q_2 \).

**Proposition 10.** \( \mathcal{L}(\Theta) = \left\{ w \in (\Sigma_1 \oplus \Sigma_2)^* : p_{\Sigma_1}(w) \in \mathcal{L}(\Theta_1) \land p_{\Sigma_2}(w) \in \mathcal{L}(\Theta_2) \right\} \).

**Proposition 11.** \( w \in \mathcal{L}(\Theta) \iff \exists w_1 \in \mathcal{L}(\Theta_1), p_{\Sigma_1}(w_1) = p_{\Sigma_1}(w) = p_{\Sigma_1 \cap \Sigma_2}(w_2) \).

From these propositions, we can characterize faulty
sequences.

**Proposition 12.** Given a system \( \Theta \) and a meta-pattern \( \Pi \):

\( w \in \mathcal{L}(\Theta) \) is faulty \iff \( w \in \mathcal{L}(\Theta \circ \Pi) \).

All necessary notations being introduced, the analysis
method is detailed in the following.

### 4.2 Analysis stages

We define an “\( \ast \)” operator applicable to sequences, alphas,
and labeled Petri nets, which renames appropriately
its operand by priming unobservable events and the label
of each place. We also change the labels of every place in
the supervision patterns \( \Omega_i \) from \( N \) (reps. \( F \)) to \( N_i \) (resp. \( F_i \)).

We start by combining the set of supervision patterns in a
meta-pattern \( \Pi = (\Omega_1 \parallel \Omega_2 \parallel \ldots \parallel \Omega_n) \) — the product
is obviously associative — then we perform the product
of \( \Theta \) and \( \Pi \) to obtain \( \Theta \Pi = \Theta \ast \Pi \). The net \( \Gamma \)
is calculated as the synchronized product of \( \Theta \Pi \) and \( \Theta' \Pi \) —
the result of this product is commonly called the twin-plant.
The final step consists in finding the possible sequences
\( c : M_{F_0} \rightarrow M \) from \( M_{F_0} \), the initial marking of \( \Gamma \) that lead
to an ambiguous marking that can indefinitely occur.

**Definition 13.** A marking \( M \) in \( \Gamma \) is ambiguous if \( \exists p, p' \in P \),
\( M(p) \times M(p') \neq 0 \) \land \( (\ell(p) = F_1 \land \ell(p') = F_2) \lor
(\ell(p) = N_1 \land \ell(p') = F_1') \).

**Definition 14.** A set of supervision patterns \( \Omega^* \) is said to
be involved in an ambiguous marking \( M \), if:

\( F_i \in \ell(M) \iff \Omega_i \in \Omega^* \).

**Definition 15.** A sequence of transitions \( c \in T_{\Gamma}^* \) is ambiguous
if it leads to an ambiguous marking.

**Definition 16.** A set of supervision patterns \( \Omega^* \) is said to
be involved in a sequence of transitions \( c \in T_{\Gamma}^* \) leading to a
marking \( M \), if \( \Omega^* \) is involved in \( M \).

We set \( Q_{F_i} = \{ q = q_1 \circ q_2 : q_1 \in \ell_{\Theta_\Pi}(F_i), q_2 \in \ell_{\Theta_\Pi}(N_i') \} \), for \( i = 1..n \).

**Proposition 17.** A reachable marking \( M \) of \( \Gamma \) is ambiguous
\( \iff \exists c \in T_{\Gamma}^* : M_{F_0} \rightarrow M \) and \( \ell(c) \in \ell(\Gamma) \).

**Definition 18.** A sequence of transitions \( c \in T_{\Gamma}^* \) is an
admissible cycle in \( \Gamma \) if \( \exists s \in T_{\Gamma}^*, \forall t \in T_{\Gamma}^* : \forall n \in N, t_{\ell}(s^n) \in \ell(\Gamma) \land c = s.t \).

From Definitions 15, 18 and Proposition 17

**Proposition 19.** \( c \) is an admissible cycle in \( \Gamma \implies c \) is an
ambiguous cycle.

The diagnosability analysis (generalization of Goung et al.
(2013)) relies on the following result.

**Theorem 20.** \( \Theta \) is not \( \Omega \)-diagnosable \iff \( \Gamma \) contains
ambiguous cycles.

The notion of discriminability proposed in this paper can be
analyzed based on the following:

**Theorem 21.** A subset \( \Omega^* \in \Omega^\Gamma \) of supervision patterns
is said to be discriminable \iff \( \Omega^* \) is not involved in any
ambiguous cycle of \( \Gamma \).

So, the problem of patterns discriminability and \( \Omega \)-
diagnosability can be reduced to finding ambiguous cycles in
\( \Gamma \).

To sum up, the method includes:
modelling the supervision patterns and the system with labeled Petri nets;
(2) compute a meta-pattern and combine it with the system;
(3) construction of a twin-plant;
(4) unfolding the labeled Petri net of the twin-plant;
(5) search for ambiguous cycles in the twin-plant;
(6) conclude on the discriminability of the patterns and consequently on the system diagnosability.

5. FINDING AMBIGUOUS CYCLES

To find cycles in a Petri net, we will use Petri net unfoldings. This technique — introduced by McMillan (1995) — attempts to avoid the combinatorial explosion inherent to the classical usage of marking graphs.

5.1 Petri net unfolding

A Petri net unfolding is another labeled Petri net that is generally infinite.

Definition 22. The unfolding $\Phi = \langle P_\Phi, T_\Phi, A_\Phi, \ell_\Phi, P, T, M_0 \rangle$ of a marked and labeled Petri net $\Theta = \langle P, T, A, \ell, L, S, M_0 \rangle$ is a labeled Petri net such that:

1. $\forall p \in P_\Phi : |p| \leq 1$.
2. $\Phi$ is acyclic, i.e. for any element $x \in P_\Phi \cup T_\Phi : \neg (x < x)$.
3. $\Phi$ is finitely preceded, i.e., for every $x \in P_\Phi \cup T_\Phi$, the set of elements $y \in P_\Phi \cup T_\Phi$ such that $y < x$ is finite.
4. $\forall (x \in P_\Phi \cup T_\Phi$ is in self-conflict $\neg (x \neq x)$.
5. $\ell_\Phi(P) \subseteq P, \ell_\Phi(T) \subseteq T$.
6. $\forall t \in T_\Phi : \ell_\Phi(t) \text{ is isomorphic to } \ell$.
7. $\forall t \in T_\Phi : \ell_\Phi(t) \text{ is isomorphic to } \ell$.
8. $M_0 = \{ x \in P_\Phi : \exists y \in T_\Phi, y < x \}$.

Where $<$ is the transitve closure of the arc relation $A_\Phi$ — called the causal relation, and $\leq$ the reflexive closure of $<$. The couple of elements $(x, y) \in (P_\Phi \cup T_\Phi)^2$ is a conflict (denoted $x \neq y$) if there exist two transitions $t_1, t_2 \in T_\Phi$, such that $t_1 \neq t_2, \ell_1 \cup \ell_2 \neq \emptyset$ and $(t_1, x), (t_2, y) \in <$.

A configuration $C$ of an unfolding is a set of causally-closed and conflict-free transitions of $T_\Phi$, formally: $\forall t \in C \Rightarrow \forall t' < t, t' \in C$ and $\forall t' C' < C$ where $C = \{ p \in \ell_\Phi, t \in C \}$ and $\ell_\Phi(C)$ characterizes a marking of the net $\Theta$. The local configuration of a transition $t \in T_\Phi$ is the set of transitions $t' \in T_\Phi$ such that $t' \leq t$.

For any bounded Petri net, McMillan (1995) shows the existence of a fragment of the unfolding $\Phi$ called the finite complete prefix of $\Phi$ appropriate to analyze a particular property of the Petri net. A partial order which leads to a complete finite prefix appropriate for detecting cycles is the strict set inclusion “$<$”.

5.2 Algorithm

Algorithm 1 looks for ambiguous cycles in the twin-plant $\Gamma$.

Algorithm 1 Search for ambiguous cycles

1: procedure ANALYSIS($\Gamma$)
2: $\Phi \leftarrow$ unfolding($\Gamma$)
3: $H \leftarrow$ cut-offs($\Phi$)
4: for all $e \in H.events$ do
5: if $\ell_\Phi(H[e])$ is ambiguous then
6: prune($H[e]$)
7: end if
8: end for
9: end procedure

Table 1. Size of intermediary Petri nets of the method

| | $|P|$ | $|T|$ |
|---|---|---|
| System ($\Theta$) | 16 | 16 |
| First pattern ($\Omega_1$) | 5 | 7 |
| Second pattern ($\Omega_2$) | 3 | 6 |
| Meta-pattern ($\Pi$) | 8 | 13 |
| Product of the system and the meta-pattern ($\Theta\Pi$) | 24 | 27 |
| Twin-plant ($\Gamma$) | 48 | 85 |
| Unfolding of the twin-plant ($\Phi$) | 82 | 38 |

We can see that even though the example is not strongly concurrent, the overhead in places and transitions is...
In this paper, we introduced the notion of discriminability of supervision patterns and proposed a method to analyse it. The discriminability of a pattern ensures that one can detect the exclusive occurrence of this particular pattern, which differs from its mere recognition without further information about other patterns. We have also shown that once the discriminability analysis is performed, we can conclude about the diagnosability of the system with no extra cost. Our approach relies on the construction of a meta-pattern embedding all interesting behaviors. After combining it with the system, we perform a twin-plant to detect ambiguous sequences by using petri net unfoldings. 

An interesting continuation of this work could be the extension of the models to include timed systems and patterns. This would permit us to monitor more complex — and interesting — behaviors that can not be modeled otherwise.

REFERENCES


