Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 51-7 (2018) 349–354

On-line Algorithm for Current State Opacity Enforcement in a Petri Net Framework

X.Y. Cong * **M.P. Fanti** ** **A.M. Mangini** *** **Z.W. Li** ****

* *School of Electro-Mechanical Engineering, Xidian University, No. 2 South Taibai Road, Xi'an 710071, China (e-mail: congxuya@163.com).*
** *Department of Electrical and Information Engineering, Polytechnic of Bari, 70125 Bari, Italy (e-mail: mariapia.fanti@poliba.it)*
*** *Department of Electrical and Information Engineering, Polytechnic of Bari, 70125 Bari, Italy (e-mail: agostinomarcello.mangini@poliba.it)*
**** *Institute of Systems Engineering, Macau University of Science and Technology, Taipa, Macau (e-mail: zhwli@xidian.edu.cn)*

Abstract: Opacity notion is a security and privacy property that verifies whether an external observer (intruder) can deduce a *secret* of a system by observing its behaviour. This paper addresses the current state opacity verification and enforcement in the framework of Petri Net (PN): an observation of the system is said to be current-state opaque if an intruder is unable to determine whether the current-state of the system belongs to a set of secret states, otherwise it is said to be not current-state opaque. The run time verifier waits for an observable event and performs an algorithm based on the solution of Integer Linear Programming problem to verify the current state opacity and preserve the secret. Indeed, if the secret may be discovered, then the last event is hidden. An example shows the efficiency of the proposed approach.

Keywords: Opacity; Petri Net; Integer Linear Programming.

1. INTRODUCTION

Security and privacy are important requirements that on-line services of networked and cyber-physical systems have to fulfill. Due to the increase of Information and Communications Technologies (ICT) systems, these issues are drawing the researchers' attention over the last few decades (Jacob et al. (2016)).

In the context of the analysis of security protocols the notion of opacity is specific to a particular item of information, called *secret*. The secret is deemed to be opaque for a particular run of a protocol if the intruder is unable to deduce its value from the observations and deductions available to him during the run.

Successively, the notion of opacity is extended to systems in general, rather than just to cryptographic protocols. Bryans et al. (2005) cast the opacity in the framework of Petri Nets (PNs): in particular, the authors are interested in: *i*) whether an observer can establish a secret at some specific state(s) of the execution of the system solely on the basis of its visible version; *ii*) finding out whether a marking, that belongs to the secret, can be deduced by the observer.

Depending on the definition of the secret, there are two main kinds of opacity notions proposed in the related literature (Jacob et al., 2016): language-based opacity and state-based opacity. State-based opacity defines the secret as a set of secret states and it can be further classified as initial-state opacity (Tong et al. (2017)), initial-and-final-state opacity (Wu and Lafortune (2013)), current-state opacity (Bryans et al. (2008), Saboori and Hadjicostis (2014), Tong et al. (2017)) and k-step opacity (Saboori and Hadjicostis (2011)), (Yin and Lafortune (2017)).

However, Wu and Lafortune (2013) establish that it is possible to transform one of the four kinds of opacity (language-based, initial-state, initial-and-final-state, and current-state) to any other by a polynomial algorithm.

This paper focuses on the verification and enforcement of current-state opacity by PNs. More in detail, an on-line algorithm is proposed to verify if a system is not current-state opaque with respect to a secret, i.e., if for any observation of finite length, the intruder cannot infer that the current state of the system belongs to the secret.

In this context, the intruder is considered as an external observer that has full knowledge of the structure of the system but has only partial observation on its events. In Saboori and Hadjicostis (2007), the authors extend the notion of opacity in computer security to Discrete Event Systems (DESs) and use a state-based approach to verify whether the system is opaque. Successively, the same authors in Saboori and Hadjicostis (2014) extend current-state opacity formulations to systems that can be modeled as probabilistic finite automata. Moreover, the paper (Chen et al. (2017)) proposes a Jensen-Shannon divergence based measure to quantify secrecy loss in systems modeled as partially observed stochastic DESs.

2405-8963
Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2018.06.324

In the framework of PNs, the notion of current-state opacity is introduced in Bryans et al. (2005). In particular, the authors assume that the system behaviour is represented by transition firings. Moreover, Bryans et al. (2008) extend to labeled transition systems the concept of current-state opacity.

Recently, Tong et al. (2017) propose an approach based on basis marking to verifying current-state opacity of bounded Labeled Petri Nets (LPNs). The advantage of this method is to avoid the exhaustive enumeration of the reachable markings. However, it can be only applied to bounded PNs and a large memory may be required.

In this paper a novel on-line verification and enforcement method of current-state opacity is developed. The authors propose an algorithm based on Integer Linear Programming (ILP) problem solutions, an approach also used to solve the on-line fault diagnosis (Basile et al. (2009), Dotoli et al. (2009) and Fanti et al. (2013)) to avoid the states enumeration of a system. More precisely, the structure of the LPN and the initial marking are known by the intruder which only has partial observation of the transitions. The run-time verifier observes the same event sequence of the intruder: it waits for an observable event and performs an algorithm to deduce whether the information is useful to determine the secret, i.e., if the observed sequence is such that the intruder can infer that all the markings consistent with the observation belong to the secret. If the secret may be discovered, then the last event is hidden by the run-time enforcer. Hence, the proposed approach falls in the opacity enforcement at run-time (Falcone and Marchand, 2015): it does not restrict the system behaviour and hides some system output events whenever it is necessary.

In this paper, the secret is defined as the conjunction of a set of Generalized Mutual Exclusion Constraints (GMECs) ((Giua et al., 1992)). Indeed, GMECs describe interesting subsets of the state space of a net and can represent many important state-based specifications and control problems (Tong et al. (2017)).

Now, we summarize the main features and contributions of this paper.

(1) An on-line algorithm for current-state opacity verification and enforcement is presented in the framework of PN system exploiting ILP problem solutions. The algorithm checks i) if the observed word (of finite length) is current-state opaque wrt the secret; ii) by using some presented propositions, the algorithm verifies if the system is not current-state opaque wrt the secret avoiding expensive off-line computations.
(2) By using the on-line strategy, the proposed method avoids the redesign and redefinition of the intruder when the system structure changes.
(3) The proposed method is general since it both can be applied to the net with bounded and unbounded state space.

The paper is organized as follows. Section 2 briefly gives some basic definitions of the PNs. Section 3 addresses the current-state opacity problem, while Section 4 first introduces the intruder specification and then, proposes an on-line algorithm to verify and enforce current-state opacity. Finally, Section 5 draws conclusion and future works.

2. BASIC DEFINITIONS ON PETRI NETS

2.1 Petri Nets

This section recalls some basic definitions on PNs used in the paper (Peterson (1981)).

Definition 1. A PN is a bipartite graph described by the four-tuple $PN = (P, T, Pre, Post)$, where P is a set of places with cardinality m represented by circles, T is a set of transitions with cardinality n represented by bars, $Pre : P \times T \to \mathbb{N}^{m\times n}$ and $Post : P \times T \to \mathbb{N}^{m\times n}$ are the pre- and post-incidence matrices, respectively, which specify the arcs connecting places and transitions. Note that $\mathbb{N}$ is the set of non-negative integers. Matrix $C = Post - Pre$ is the incidence matrix of the PN.

More precisely, for each $p \in P$ and $t \in T$ element $Pre(p,t)$ ($Post(p,t)$) is equal to a natural number indicating the arc multiplicity if an arc going from p to t (from t to p) exists, and it is equal to 0 otherwise.

The state of a PN is represented by its current marking that is a mapping $M : P \to \mathbb{N}^m$, assigning to each place a non-negative number of tokens. The marking of place p is denoted by $M(p)$. A PN system $\langle PN, M_0\rangle$ is a net PN with an initial marking M_0.

For the pre- and post-sets we use the dot notation, e.g., ${}^\bullet t = \{p \in P : Pre(p,t) > 0\}$. A transition $t_j \in T$ is enabled at marking M if and only if for each $p \in {}^\bullet t_j$ it holds: $M \geq Pre(p, t_j)$ and $M[t_j\rangle$ is written to denote that $t_j \in T$ is enabled at M.

When the transition t_j fires, it produces a new marking M_{new}, denoted by $M[t_j\rangle M_{new}$ that is computed by the PN state equation $M_{new} = M + C \cdot \overrightarrow{t_j}$, where $\overrightarrow{t_j}$ is an n-dimensional firing vector corresponding to the j-th canonical basis vector.

Let $\sigma = t_{\alpha_1} t_{\alpha_2} \dots t_{\alpha_k}$ be a sequence of transitions (firing sequence) and let k be its length, given by the number of transitions that σ contains. The fact that a transition $t \in T$ appears in the sequence σ is denoted by $t \in \sigma$. Moreover, the notation $M[\sigma\rangle$ denotes that σ is enabled at M and $M[\sigma\rangle M_{new}$ denotes that the firing of σ yields M_{new}. The set of all sequences that can fire in a net system $\langle PN, M_0\rangle$ is denoted by $L(PN, M_0) = \{\sigma \in T^* \mid M_0[\sigma\rangle\}$. In addition, we define $\boldsymbol{\sigma} : T \to \mathbb{N}^n$ the firing vector associated with σ.

A marking M is reachable from $\langle PN, M_0\rangle$ if there exists a firing sequence σ such that $M_0[\sigma\rangle M$. The set of all markings reachable from M_0 defines the reachability set of $\langle PN, M_0\rangle$ denoted as $R(PN, M_0)$.

A PN having no directed cycles is said to be *acyclic*. The following theorem shows an important result for this subclass of PN.

Theorem 2. (Ichikawa and Hiraishi (1988)) Let $\langle PN, M_0\rangle$ be an acyclic PN.

If vector $\boldsymbol{y}$ satisfies equation $M_0 + C \cdot \boldsymbol{y} \geq \mathbf{0}$, there exists a firing sequence σ fireable from M_0 such that $\boldsymbol{\sigma} = \boldsymbol{y}$.

A marking M is reachable from M_0 iff there exists a non-negative integer solution $\boldsymbol{y}$ satisfying the state equation $M = M_0 + C \cdot \boldsymbol{y}$.

2.2 Labeled Petri Nets

Definition 3. An LPN is a four-tuple $G = (PN, M_0, E, \lambda)$ where $\langle PN, M_0 \rangle$ is a PN system, E is an alphabet (a set of labels) and $\lambda : T \to E \cup \{\varepsilon\}$ is a labeling function that assigns to each transition $t \in T$ either a symbol $e \in E$ or the empty word ε.

Namely, the set of transitions can be partitioned into $T = T_o \cup T_u$ with $T_o \cap T_u = \emptyset$, where T_o (resp. T_u) is the set of $|T_o| = n_o$ (resp. $|T_u| = n_u$) observable (resp. unobservable) transitions.

The labeling function λ is defined as follows: if $t \in T_o$ then $\lambda(t) = e \in E$, and if $t \in T_u$ then $\lambda(t) = \varepsilon$. In this paper, we assume that the same label $e \in E$ can be associated to more than one transition. In the following, we denote by $T(e) = \{t \in T_o | \lambda(t) = e\}$ the set of transitions associated with the same label $e \in E$.

Moreover, we denote as w the sequence of events associated with the sequence $\sigma \in T^*$ such that $w = \lambda(\sigma)$ by using the extended form of the labeling function $\lambda : T^* \to E^*$. The set of languages generated by an LPN is denoted as $\mathcal{L}(PN, M_0) = \{w \in E^* | \exists \sigma \in L(PN, M_0) : \lambda(\sigma) = w\}$. In addition, we denote by $\sigma_u \in \sigma$ ($\sigma_o \in \sigma$) the subsequence of σ composed of the unobservable (observable) transitions and by $\boldsymbol{\sigma}_u : T_u \to \mathbb{N}^{n_u}$ ($\boldsymbol{\sigma}_o : T_o \to \mathbb{N}^{n_o}$) the corresponding firing vector.

Given a net $PN = (P, T, Pre, Post)$ and a subset $T_A \subseteq T$ of its transitions, we define the T_A-induced subnet of PN as a new net $PN_A = (P, T_A, Pre_A, Post_A)$ where Pre_A and $Post_A$ are the restrictions of Pre and $Post$ to T_A, i.e., PN_A is the net obtained from PN by removing all transitions in $T \backslash T_A$, which is denoted by $PN_A \angle_{T_A} PN$. In the following, matrices $C_u = Post_u - Pre_u$ and $C_o = Post_o - Pre_o$ denote the restriction of the incidence matrix C to T_u and T_o, respectively.

Let w be an observed word. We define $\mathcal{S}(w) = \{\sigma \in L(PN, M_0) | \lambda(\sigma) = w\}$ as the set of firing sequences consistent with w and $\mathcal{C}(w) = \{M \in \mathbb{N}^m | \exists \sigma \in \mathcal{S}(w) : M_0[\sigma\rangle M\}$ as the set of markings consistent with w.

3. CURRENT-STATE OPACITY PROBLEM

In this section, we provide a description of the current-state opacity problem.

We assume that the intruder has complete knowledge of the net system but partial observation of its behaviour. In particular, the intruder can detect the occurrence of all the observable transitions $t \in T_o$ only. Moreover, the secret is defined as a set of secret states $S \subseteq R(PN, M_0)$.

In the following, we first recall opacity definitions reported in Tong et al. (2017) for DESs modeled by LPNs.

Definition 4. (Tong et al. (2017)). Let G be an LPN system and S be a secret. An observation w of G is said to be current-state opaque wrt S if $\mathcal{C}(w) \nsubseteq S$ holds.

Based on Definition 4, the current-state opacity definition for a system is given as follows.

Definition 5. (Tong et al. (2017)). Let G be an LPN system and S be a secret. G is said to be current-state opaque wrt S if all observations w are current-state opaque wrt S.

Motivated by Definitions 4 and 5, we provide the following two definitions of a not current-state opaque observation and a not current-state opaque system, respectively.

Definition 6. Let G be an LPN system and S be a secret. An observation w of G is said to be not current-state opaque wrt S if $\mathcal{C}(w) \subseteq S$ holds.

A not current-state opaque observation w implies that the intruder can infer that all the markings consistent with w belong to the secret, i.e., $\forall M \in \mathcal{C}(w) : M \in S$.

Consequently, a not current-state opaque system is defined as follows.

Definition 7. Let G be an LPN system and S be a secret. G is said to be not current-state opaque wrt S if there exists an observation w that is not current-state opaque wrt S.

In this paper, the following set of GMECs describes the secret:

$$S = \bigcap_{q=1}^{r} \{M \in \mathbb{N}^m | x_q^T \cdot M \leq k_q\},$$

where $x_q \in \mathbb{Z}^m$ and $k_q \in \mathbb{Z}$ with $q = 1, 2, \ldots, r$. Note that $\mathbb{Z}$ is the set of integers. Such a set of GMECs (x_q, k_q) is denoted as $S = \{M \in \mathbb{N}^m | X \cdot M \leq K\}$, where $X = [x_1, x_2, \ldots, x_r]^T$ and $K = [k_1, k_2, \ldots, k_r]^T$.

4. ON-LINE ALGORITHM FOR CURRENT STATE OPACITY ENFORCEMENT

In this section we provide an on-line approach for verification and enforcement of current-state opacity by a centralized approach. First, we introduce the intruder specification and then, we describe in detail the on-line algorithm that verifies and enforces current-state opacity.

4.1 Intruder Specification

Given an observed word w, we show how to characterize the sets $\mathcal{S}(w)$ and $\mathcal{C}(w)$ by solving ILP problems and we specify the on-line intruder. Firstly, the following assumption is given for the system under investigation:

A1) The T_u-induced subnet $PN_u \angle_{T_u} PN$ and T_o-induced subnet $PN_o \angle_{T_o} PN$ are acyclic.

In particular, assumption A1 allows us to study the reachability of the unobservable and observable subnets by using the state equation. The inputs of the intruder are the

LPN system $G = (PN, M_0, E, \lambda)$, the secret S modeled by a set of GMECs, and the observed word $w \in \mathcal{L}(PN, M_0)$. The output of the intruder is the set-valued function $\Phi(w)$ that is defined as follows:

Definition 8. An on-line intruder is a function $\Phi : \mathcal{L}(PN, M_0) \to \{Y, N\}$ that associates to each observation $w \in \mathcal{L}(PN, M_0)$ the following sets:

(1) $\Phi(w) = \{Y\}$ if the observation of the system is current-state opaque wrt the secret S.
(2) $\Phi(w) = \{N\}$ if the observation of the system is not current-state opaque wrt the secret S.

Given an LPN system $G = (PN, M_0, E, \lambda)$ with language $\mathcal{L}(PN, M_0)$ and satisfying assumption A1, we specify an intruder that works on-line and determines whether an observation is opaque or not after the occurrence of each new event. More precisely, for each initial marking $M_0 \in \mathbb{N}^m$, at the occurrence of an observed word $w \in \mathcal{L}(PN, M_0)$, the following proposition shows a linear algebraic characterization of each transition sequence $\sigma \in T^*$ whose firing at M_0 is consistent with the observation $w = \lambda(\sigma)$.

Proposition 9. Let $\mathcal{L}(PN, M_0)$ be the language of an LPN system $G = (PN, M_0, E, \lambda)$ satisfying assumption A1. Given a word $w \in \mathcal{L}(PN, M_0)$ denoted by $w = e_1 e_2 \dots e_h$ (where $e_i \in E$ for $i = 1, 2, \dots, h$ is the ith observed event), there exists at least one sequence $\sigma = \sigma_{u_1}\sigma_{o_1}\sigma_{u_2}\sigma_{o_2}\dots\sigma_{u_h}\sigma_{o_h}\sigma_{u_{h+1}}$ with $|\sigma_{u_i}| \geq 0$ for $i = 1, 2, \dots, h+1$ and $|\sigma_{o_i}| = 1$ for $i = 1, 2, \dots, h$ enabled at the initial marking M_0 such that $\lambda(\sigma) = w = e_1 e_2 \dots e_h$ *iff* there exist $2h+1$ firing vectors $\boldsymbol{\sigma}_{u_1}, \boldsymbol{\sigma}_{u_2}, \dots, \boldsymbol{\sigma}_{u_{h+1}}, \boldsymbol{\sigma}_{o_1}, \boldsymbol{\sigma}_{o_2}, \dots \boldsymbol{\sigma}_{o_h}$ that satisfy the following set of constraints denoted by $\rho(M_0, w)$:

$$
\begin{cases}
\boldsymbol{\sigma}_{u_i} \in \mathbb{N}^{n_u}, \quad for\ i = 1, \dots, h+1 & (a)\\
\boldsymbol{\sigma}_{o_i} \in \mathbb{N}^{n_o}, \quad for\ i = 1, \dots, h & (b)\\
C_u \sum_{i=1}^{k} \boldsymbol{\sigma}_{u_i} \geq Pre_o \cdot \boldsymbol{\sigma}_{o_k} - M_0 - C_o \sum_{i=1}^{k-1} \boldsymbol{\sigma}_{o_i}, & \\
for\ k = 1, \dots, h & (c)\\
M_0 + C_u \sum_{i=1}^{h+1} \boldsymbol{\sigma}_{u_i} + C_o \sum_{i=1}^{h} \boldsymbol{\sigma}_{o_i} \geq \mathbf{0} & (d)\\
\sum_{t_j \in T(e_1)} \boldsymbol{\sigma}_{o_1}(t_j) = 1 & \\
\sum_{t_j \in T(e_2)} \boldsymbol{\sigma}_{o_2}(t_j) = 1 & \\
\dots & \\
\sum_{t_j \in T(e_h)} \boldsymbol{\sigma}_{o_h}(t_j) = 1 & (e)\\
\sum_{t_j \notin T(e_1)} \boldsymbol{\sigma}_{o_1}(t_j) = 0 & \\
\sum_{t_j \notin T(e_2)} \boldsymbol{\sigma}_{o_2}(t_j) = 0 & \\
\dots & \\
\sum_{t_j \notin T(e_h)} \boldsymbol{\sigma}_{o_h}(t_j) = 0 &
\end{cases}
\tag{1}
$$

Proof. The proof is omitted.

In general, the solution of the set of constraints $\rho(M_0, w)$ is not a singleton and fully characterizes the two sets $\mathcal{S}(w)$ and $\mathcal{C}(w)$. Actually, constraints (1) imply that $M = M_0 + C_u \sum_{i=1}^{h+1} \boldsymbol{\sigma}_{u_i} + C_o \sum_{i=1}^{h} \boldsymbol{\sigma}_{o_i}$ belongs to $\mathcal{C}(w)$. In order to verify if the behaviour of the system remains in the secret under the given observation $w \in \mathcal{L}(PN, M_0)$, we have to find a possible solution of (1), i.e., a set of firing vectors leading to a marking that does not belong to the secret.

Algorithm specifying the on-line verifier
Input: $G = (PN, M_0, E, \lambda)$, S
Output: $\Phi(w)$
Step 1. *Initializing the variables of the algorithm*
$w := \varepsilon$, $h := 0$,
Step 2. *Recording the events*
Wait until an event $e \in E$ occurs
$w := we$, $\Phi(w) := \emptyset$ $h := h + 1$.
Step 3. *Verifying if the observed word w is current-state opaque wrt S*
for $q = 1$ to r **do**
Solve ILPP 1
if $z_q > k_q$ **then**
set $\Phi(w) := \{Y\}$, **go to Step 2**
end if
end for
Step 4. set $\Phi(w) := \{N\}$, **hide** event e, **go to Step 2**.

The following theorem proves that such a solution can be obtained by solving the ILP Problem 1 (ILPP 1).

Proposition 10. Let $G = (PN, M_0, E, \lambda)$ be an LPN system and S be a secret. Given an observed word $w = e_1 e_2 \dots e_h \in \mathcal{L}(PN, M_0)$, let us define the following ILP problem, ILPP 1:

$$
\begin{cases}
z_q = max & x_q^T \cdot M\\
s.t. & \rho(M_0, w)\\
& M = M_0 + C_u \sum_{i=1}^{h+1} \boldsymbol{\sigma}_{u_i} + C_o \sum_{i=1}^{h} \boldsymbol{\sigma}_{o_i}.
\end{cases}
\tag{2}
$$

An observation w of G is current-state opaque wrt S *iff* for a GMEC (x_q, k_q) of the secret, ILPP 1 admits a solution $\boldsymbol{\sigma}_{u_1}, \boldsymbol{\sigma}_{u_2}, \dots, \boldsymbol{\sigma}_{u_{h+1}}, \boldsymbol{\sigma}_{o_1}, \boldsymbol{\sigma}_{o_2}, \dots, \boldsymbol{\sigma}_{o_h}$ and it holds $z_q > k_q$.

Proof. The proof is omitted.

Proposition 10 provides the following sufficient and necessary condition to verify whether an LPN system is not current-state opaque.

Corollary 11. Let $G = (PN, M_0, E, \lambda)$ be an LPN system and S be a secret. The system is not current-state opaque *iff* there exists an observation w such that ILPP 1 admits a solution with $z_q = \max x_q^T \cdot M \leq k_q$ for each GMEC (x_q, k_q) of the secret.

Proof. It follows immediately from Definition 7 and Proposition 10. □

4.2 On-line Algorithm

Based on the aforementioned results, we propose Algorithm 4.2 that the verifier applies on-line by observing the nsame events of the intruder to verify the current-state opacity of an observation of a given LPN system. Moreover, if there exists an observation w such that ILPP 1 admits a solution with $z_q = \max x_q^T \cdot M \leq k_q$ for each GMEC (x_q, k_q) of the secret, then the system is not current-state opaque. In the following, we discuss the details of Algorithm 4.2.

Step 1 initializes the variables of the algorithm where h denotes the length of w. Step 2 waits for a new observed

event, while Step 3 verifies whether w is current-state opaque wrt S. If there exists a GMEC (x_q, k_q) of S such that the objective function value of ILPP 1 $z_q > k_q$, then by Proposition 10, w is current-state opaque wrt S. In this case, the algorithm goes to Step 2 to wait for a new event. Moreover, if ILPP 1 admits an optimal solution with $z_q \leq k_q$ for each of the GMEC (x_q, k_q) of the secret, then by Proposition 10, w is not current-state opaque wrt S. According to Corollary 11, the system G is not current-state opaque wrt S: then the enforcer hides the last event $e \in E$.

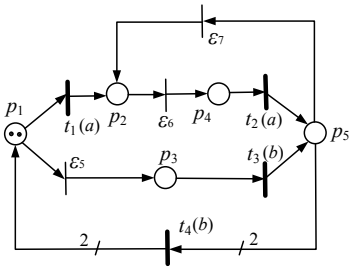

Fig. 1. The LPN system considered in Example 12.

Example 12. In order to show the application of Algorithm 4.2, we consider the LPN of a communication system proposed in Tong et al. (2017) that is shown in Fig. 1. Let us consider the LPN in Fig. 1. There are five places, seven transitions and two observable events, i.e., $E = \{a, b\}$. The set of observable transitions is $T_o = \{t_1, t_2, t_3, t_4\}$ such that $T(a) = \{t_1, t_2\}$ and $T(b) = \{t_3, t_4\}$, and the set of unobservable transitions is $T_u = \{\varepsilon_5, \varepsilon_6, \varepsilon_7\}$. The initial marking is $M_0 = [2, 0, 0, 0, 0]^T$. Let the secret be $S = \{M \in \mathbb{N}^5 | X \cdot M \leq K\}$, where $X = [x_1, x_2]^T$ and $K = [k_1, k_2]^T$ with $x_1 = [-1, 0, 1, 0, 1]$, $x_2 = [0, 0, 1, 0, 0]$, $k_1 = 1$ and $k_2 = 0$. By using the Basis Reachability Graph (BRG) and its observer presented in Tong et al. (2017), the system is inferred not current-state opaque wrt the secret. Now, we show the procedure of the proposed on-line algorithm as follows.

Assume that the observable event a occurs. By applying Algorithm 4.2, we infer $\Phi(a) = \{Y\}$, i.e., the observation $w = a$ is current-state opaque wrt S.

Now, assume that the second observable event b occurs: Algorithm 4.2 provides $\Phi(ab) = \{N\}$, i.e., the observation $w = ab$ is not current-state opaque wrt S and the second event occurrence b is hidden. Moreover, according to Definition 7, the system is not current-state opaque wrt S. □

4.3 Computational Complexity

A global picture of complexity results for the verification of diagnosability and opacity is discussed in (Brard et al. (2017)). As regard the computational complexity of Algorithm 4.2, we note that the algorithm needs to solve for each observation w at most r ILPPs, which are NP-hard in theory. To evaluate the computational effort required by the proposed algorithm, we recall that the primary determinants of the computational cost of an ILPP are the numbers of variables and constraints in it. It is easy to infer that the numbers of variables and constraints in each ILPP are $h \cdot n + n_u + m$ and $h \cdot m + 2 \cdot (h + m)$ in the worst case, respectively, where $h \geq 0$ denotes the length of the observation w, n denotes the number of transitions in the LPN, n_u denotes that of unobservable transitions in the LPN and m denotes that of places in the LPN. Hence, the on-line computational cost of the proposed algorithm increases with the number of observed events. However, in practice, our experience shows that in the examined cases, compared with those presented in the literature, an optimal solution is obtained in a short time by solving the ILPPs on a PC equipped with a standard solver of optimization tool.

4.4 Experimental Results

This subsection provides some experimental results of the algorithm proposed in this paper. The obtained computational time refers to the CPU seconds of a notebook computer under the Windows 7 operating system with Intel CPU Core 2.6 GHz, 8 GB memory and a standard optimization solver.

In order to show the advantage and efficiency of the proposed on-line algorithm, let us consider a large example satisfying Assumption A1 shown in Fig. 2, which is similar to the one presented in Cabasino et al. (2011). This example is an LPN that models a manufacturing system. The LPN is composed by 35 places, 28 transitions and the event set is $E = \{a, b, c, d\}$. The set of observable transitions T_o consists of transitions from t_1 to t_{10} such that $T(a) = \{t_1, t_4\}$, $T(b) = \{t_2, t_3, t_9\}$, $T(c) = \{t_5, t_6\}$, $T(d) = \{t_7, t_8, t_{10}\}$. The secret is defined by the following set: $S = \{M \in \mathbb{N}^{35} | M(p_{12}) + M(p_{15}) + M(p_{21}) + M(p_{22}) \leq 0\}$. By applying the method in Tong et al. (2017) to this example, the system is not current-state opaque wrt the secret. However, it takes more than 1800 seconds to obtain this result due to the computation of the BRG and its observer.

Now, we apply Algorithm 4.2 to this example. In particular, the performance of Algorithm 4.2 is presented in Table 1, where the first column represents the evolution of the system, N_{var} and N_{con} indicate the numbers of variables and constraints of ILPP 1, respectively. The fourth column shows the CPU time in seconds for solving ILPP 1 and the fifth column is the output of Algorithm 4.2 at each step. From Table 1, we can see that the observed event sequence $w = accabcc$ is not current-state opaque wrt the secret S. Hence, according to Definition 7, we conclude that the LPN system is not current-state opaque wrt the secret.

Table 1. Performance of Algorithm 4.2

Action	N_{var}	N_{con}	time (s)	$\Phi(w)$
observable event a occurs	81	107	0.8×10^{-2}	{Y}
observable event c occurs	109	144	1.1×10^{-2}	{Y}
observable event c occurs	137	181	1.2×10^{-2}	{Y}
observable event a occurs	165	218	1.5×10^{-2}	{Y}
observable event b occurs	193	255	1.7×10^{-2}	{Y}
observable event c occurs	221	292	1.9×10^{-2}	{Y}
observable event c occurs	249	329	2.4×10^{-2}	{N}

5. CONCLUSION

This paper proposes an on-line verification method of current-state opacity in a LPN framework. The verifier

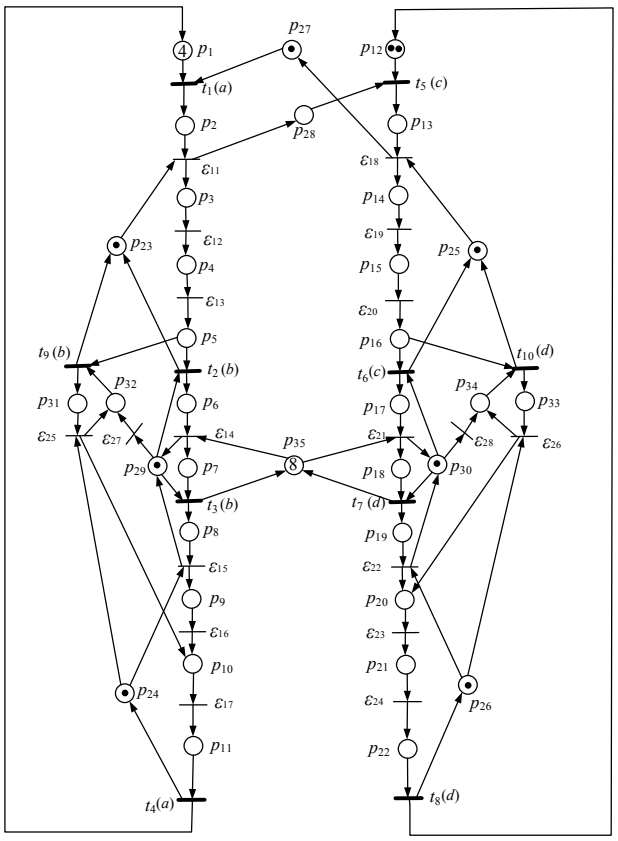

Fig. 2. LPN modeling a manufacturing system.

observes and stores the event sequence of the LPN system and decides run-time whether the given observation is current-state opaque or not. To this aim, an ILP problem is defined and we prove that, on the basis of the provided solutions at each observed event, it is possible to decide if the secret can be discovered. If the given observation of the system is not current-state opaque, the last event occurrence is hidden by the enforcer.

The proposed methodology for opacity verification falls in the opacity enforcement at run-time approach (Falcone and Marchand, 2015) that avoids the enumeration of the PN markings and can be applied also to unbounded nets. Moreover, the presented run-time enforcement is not intrusive and does not distort the internal system behaviour. By applying the proposed algorithm to a large example, we show its efficiency.

Future works will focus on the possibility of extending the proposed approach to systems where the secret cannot be represented by a set of GMECs.

REFERENCES

Basile, F., Chiacchio, P., and Tommasi, G.D. (2009). An efficient approach for online diagnosis of discrete event systems. *IEEE Transactions on Automatic Control*, 54(4), 748–759.

Bryans, J.W., Koutny, M., Mazaré, L., and Ryan, P.Y.A. (2008). Opacity generalised to transition systems. *International Journal of Information Security*, 7(6), 421–435.

Bryans, J.W., Koutny, M., and Ryan, P.Y. (2005). Modelling opacity using Petri nets. *Electronic Notes in Theoretical Computer Science*, 121, 101 – 115.

Brard, B., Haar, S., Schmitz, S., and Schwoon, S. (2017). The complexity of diagnosability and opacity verification for Petri nets. *Application and Theory of Petri Nets and Concurrency. PETRI NETS 2017*, 10258, 531–570.

Cabasino, M., Giua, A., Pocci, M., and Seatzu, C. (2011). Discrete event diagnosis using labeled Petri nets. an application to manufacturing systems. *Control Engineering Practice*, 19(9), 989 – 1001.

Chen, J., Ibrahim, M., and Kumar, R. (2017). Quantification of secrecy in partially observed stochastic discrete event systems. *IEEE Transactions on Automation Science and Engineering*, 14(1), 185–195.

Dotoli, M., Fanti, M.P., Mangini, A.M., and Ukovich, W. (2009). On-line fault detection in discrete event systems by Petri nets and integer linear programming. *Automatica*, 45(11), 2665 – 2672.

Falcone, Y. and Marchand, H. (2015). Enforcement and validation (at runtime) of various notions of opacity. *Discrete Event Dynamic Systems*, 25(4), 531–570.

Fanti, M.P., Mangini, A.M., and Ukovich, W. (2013). Fault detection by labeled Petri nets in centralized and distributed approaches. *IEEE Transactions on Automation Science and Engineering*, 10(2), 392–404.

Giua, A., DiCesare, F., and Silva, M. (1992). Generalized mutual exclusion contraints on nets with uncontrollable transitions. In *[Proceedings] 1992 IEEE International Conference on Systems, Man, and Cybernetics*, 974–979 vol.2.

Ichikawa, A. and Hiraishi, K. (1988). Analysis and control of discrete event systems represented by Petri nets. In P. Varaiya and A.B. Kurzhanski (eds.), *Discrete Event Systems: Models and Applications*, 115–134. Springer Berlin Heidelberg, Berlin, Heidelberg.

Jacob, R., Lesage, J.J., and Faure, J.M. (2016). Overview of discrete event systems opacity: Models, validation and quantification. *Annual Review in Control*, 41, 135 – 146.

Peterson, J. (1981). *Petri net theory and the modeling of systems*. Foundations of Philosophy Series. Prentice-Hall.

Saboori, A. and Hadjicostis, C.N. (2007). Notions of security and opacity in discrete event systems. In *2007 46th IEEE Conference on Decision and Control*, 5056–5061.

Saboori, A. and Hadjicostis, C.N. (2011). Verification of k -step opacity and analysis of its complexity. *IEEE Transactions on Automation Science and Engineering*, 8(3), 549–559.

Saboori, A. and Hadjicostis, C.N. (2014). Current-state opacity formulations in probabilistic finite automata. *IEEE Transactions on Automatic Control*, 59(1), 120–133.

Tong, Y., Li, Z., Seatzu, C., and Giua, A. (2017). Verification of state-based opacity using Petri nets. *IEEE Transactions on Automatic Control*, 62(6), 2823–2837.

Wu, Y.C. and Lafortune, S. (2013). Comparative analysis of related notions of opacity in centralized and coordinated architectures. *Discrete Event Dynamic Systems*, 23(3), 307–339.

Yin, X. and Lafortune, S. (2017). A new approach for the verification of infinite-step and k-step opacity using two-way observers. *Automatica*, (June), 531–570.