Symbolic Verification of Current-state Opacity of Discrete Event Systems Using Petri Nets

Journal:	*IEEE Transactions on Systems, Man and Cybernetics: Systems*
Manuscript ID	SMCA-20-12-2603
Manuscript Type:	Regular Paper
Date Submitted by the Author:	14-Dec-2020
Complete List of Authors:	Dong, Yifan; Macau University of Science and Technology Li, Zhiwu; Xidian University, School of Electro-Mechanical Engineering Wu, Naiqi; Institute of Systems Engineering, Macau University of Science and Technology
Key Words:	Discrete event systems, Petri nets, Control systems

Symbolic Verification of Current-state Opacity of Discrete Event Systems Using Petri Nets

Yifan Dong, Zhiwu Li, *Fellow, IEEE*, and Naiqi Wu, *Fellow, IEEE*

***Abstract*—Given a discrete event system, it is said to be opaque if an intruder who can partially observe system behavior cannot infer that the system is necessarily in a predefined secret. A discrete event system is current-state opaque if an intruder is never able to verify whether the current state of the system is within the secret states. This paper addresses the verification of current-state opacity of a discrete event system modeled with Petri nets, where the secret is defined as a set of states. The existing methods cannot deal with large-sized systems due to the curse of dimensionality. Multi-valued decision diagrams are widely accepted as an efficient and compact data structure for representing the extremely large reachability sets of Petri nets. We show that multi-valued decision diagrams can be used in Petri nets with partial observable transitions and propose an approach for the verification of current-state opacity for bounded Petri nets. It is shown by experimental studies that the symbolic approach is practically efficient than traditional techniques such as basis reachability graphs.**

***Index Terms*—Discrete event system, Opacity, Petri net, Multi-valued decision diagram.**

I. Introduction

SECURITY is a crucial property in cyber-infrastructures, ranging from Internet and mobile communication to various computer systems. Various notions pertaining to security are formulated such as non-interference [1], anonymity [2], and opacity [3]–[7]. In this paper, we focus on the verification of an opacity property that hides a secret from unauthorized people (called intruders). Opacity can be classified as state-based opacity and language-based opacity, while the former includes initial-state opacity, and current-state opacity [8]–[11]. This paper touches upon the current-state opacity.

Current-state opacity defines the secret of a system as a set of markings (states) when modeled by Petri nets with only partial observable transitions [8], [11], [12]. A system is said to be current-state opaque if the intruder cannot determine whether the current state belongs to the given secret. In other words, given a secret described by a subset of the reachability set and a state estimation generated by any possible sequence of observable transitions, the intruder can never be able to infer whether the current state estimation is fully included in the set of secret states.

* This work is supported by the Science Technology Development Fund, MSAR, under Grant No. 0012/2019/A1. *(Corresponding author: Zhiwu Li)*

Yifan Dong is with the Institute of Systems Engineering, Macau University of Science and Technology, Taipa, Macau, China (e-mail: 2009853UMI30001@student.must.edu.mo).

Zhiwu Li and Naiqi Wu are with the Institute of Systems Engineering, Macau University of Science and Technology, Taipa, Macau, China (email: zwli@must.edu.mo, nqwu@must.edu.mo).

Traditionally, the most intuitive method for the verification of current-state opacity of a system modeled with labeled Petri nets is to construct a reachability graph initially, and then convert it into a deterministic finite automaton, which has complexity of $\mathcal{O}(2^N)$, where N is the size of the reachability graph of a Petri net [3]. To mitigate the state explosion problem for the construction of a reachability graph, the notions of basis markings and explanations are proposed for the standard labeled Petri nets (LPNs), i.e., the Petri nets whose transitions are assigned labels and divided into unobservable and observable classes, which are introduced in [13]–[19] to solve the problems of state estimation, fault diagnosis, diagnosability analysis, reachability analysis and supervisory control.

Tong *et al*. [8], [11] use the notions of basis markings and explanations in LPNs, where a reachability graph is compactly represented by a basis reachability graph (BRG) under the assumption that the unobservable subnet is acyclic. In this case, the time complexity of the conversion from a non-deterministic finite automaton to a deterministic finite automaton can be reduced from $\mathcal{O}(2^N)$ to $\mathcal{O}(2^{N_b})$, with N_b being the number of basis markings, where in general $N_b \ll N$. In an LPN system with a small number of unobservable transitions, the number of basis markings can be also large [20]. Then, it is hard to construct the observer of the BRG due to the time complexity if a real system is large-sized.

Traditional approaches for the construction of an observer are explicit, i.e., the reachable states of Petri nets are generated in a brute-force way and stored in different and independent memory location [21], [22]. The explicit representation for a reachability set would impose a serious restriction on Petri nets with a large number of states. Recent advancements in reachability set generation based on implicit (symbolic) techniques are quite promising. The implicit method means that a set of states can be stored in a memory location, while the reachability set can be generated concurrently.

Bryant *et al*. [23], [24] provide a method based on Binary Decision Diagrams for Boolean functions, but this approach mainly focuses on safe Petri nets. The authors in [25] propose another way for non-safe Petri nets using binary decision diagrams by encoding the integer number of tokens in a place into binary variables; however, it may complicate the computation of transition relations. Alternative approaches to handle non-safe Petri nets are developed in [26], where Multi-valued Decision Diagrams (MDDs) and Kronecker representations are used to describe the reachability set and transition relations, respectively. In an improved method [27], [28], the notion of matrix diagrams for transition relations of Petri nets is proposed.

In [29] and [30], a more efficient traversal method based on the notion of saturation for the computation of state space is proposed, which is an iteration strategy by separating the events according to their affected submodels. The MDD-based approach for reachable state computation makes it possible for compactly storing a set of states of Petri nets and provides a mechanism to compute the next set of states concurrently [26], [29]–[32]. We believe that the approach based on MDDs can be employed to efficiently address the current-state opacity problem.

Based on the work in [8], [11], and motivated by the inefficiency of the existing related studies using LPNs, we construct in this paper an MDD-based observer and a verifier. The latter can be used to verify current-state opacity of a system efficiently. Compared with the traditional methods of generating the observers based on a reachability graph or a BRG, the proposed approach requires no construction of the two graphs. Furthermore, it is applicable to more general LPNs, i.e., the unobservable subnet could be either acyclic or cyclic.

This paper is structured as follows. The notions of Petri nets are recalled in Section 2. Some operation rules about MDDs are introduced in Section 3. In Section 4, we design an MDD-based approach to verify current-state opacity. In Section 5, numerical examples that illustrate the proposed approach are reported. Finally, Section 6 concludes the paper and discusses future work.

II. Preliminary

A. Petri Net

A Petri net is a structure $N = (P, T, Pre, Post)$, where P is a set of m places and T is a set of n transitions. $Pre : P \times T \to \mathbb{N}$ and $Post : P \times T \to \mathbb{N}$ are the pre- and post-incidence functions that respectively specify the arcs directed from places to transitions, and vice versa (in this work, we use $\mathbb{N}$ to denote the set of non-negative integers). Function Pre ($Post$) can be tabulated in a rectangular array and further represented by an $m \times n$ matrix indexed by P and T. We use $C = Post - Pre$ to denote the incidence matrix of a Petri net.

A marking (or state) of a Petri net is defined as a mapping $M : P \to \mathbb{N}$ that assigns to each place of the Petri net a non-negative integer number of tokens, which is graphically represented by black dots. A marking M can be represented by a column vector indexed by P. We denote $M(p)$ as the number of tokens in place p, and for economy of space, denote a marking M as $M = \sum_{p \in P} M(p) \cdot p$, i.e., the sum of all places with the token number as their coefficients. A Petri net system $\langle N, M_0 \rangle$ is a net structure N with an initial marking M_0.

A transition t is enabled at a marking M if $M \geq Pre(\cdot, t)$ and may fire yielding a new marking $M' = M + C(\cdot, t)$ (denoted as $M[t\rangle M'$). We write $M[\sigma\rangle$ to denote that a sequence of transitions $\sigma = t_{j1} \cdots t_{jk}$ is sequentially enabled at M, and $M[\sigma\rangle M'$ to denote that the firing of σ yields M'. A marking M' is reachable from marking M if there exists a sequence of transitions σ such that $M[\sigma\rangle M'$, and the set of all markings reachable from M_0 is denoted by $R(N, M_0)$, i.e., $R(N, M_0) = \{M \in \mathbb{N}^m | \exists \sigma \in T^* : M_0[\sigma\rangle M\}$. If there exists a non-negative integer $k \in \mathbb{N}$ such that for all places $p \in P$ and all reachable markings $M \in R(N, M_0)$, $M(p) \leq k$ holds, the net system $\langle N, M_0 \rangle$ is said to be bounded. A Petri net is said to be acyclic if there are no oriented cycles.

B. Labeled Petri Net

A labeled Petri net (LPN) is a four-tuple $G = (N, M_0, \Sigma, l)$, where $\langle N, M_0 \rangle$ is a Petri net system, Σ is an alphabet, and $l : T \to \Sigma \cup \{\varepsilon\}$ is a labeling function assigning to each transition $t \in T$ either a symbol from Σ or the empty word ε. The transitions in G can be divided into two disjoint sets $T = T_o \cup T_u$, where $T_o = \{t \in T \,|\, l(t) \in \Sigma\}$ is the set of observable transitions and $T_u = T \backslash T_o = \{t \in T \,|\, l(t) = \varepsilon\}$ is the set of unobservable transitions. The labeling function can be extended to transition sequences $l : T^* \to \Sigma^*$, i.e., $l(\sigma t) = l(\sigma) l(t)$ with $\sigma \in T^*$ and $t \in T$.

Given an LPN $G = (N, M_0, \Sigma, l)$ and a marking $M \in R(N, M_0)$, we define the language generated from M as

$$\mathcal{L}(N, M) = \{w \in \Sigma^* | \exists \sigma \in T^* : M[\sigma\rangle \text{ and } l(\sigma) = w\}.$$

A string $w \in \mathcal{L}(N, M_0)$ is called an observation. Let w be an observation of an LPN $G = (N, M_0, \Sigma, l)$. The set of markings consistent with w is defined as

$$\mathcal{C}(w) = \{M \in \mathbb{N}^m | \exists \sigma \in T^* : M_0[\sigma\rangle M \text{ and } l(\sigma) = w\}.$$

III. Multi-valued Decision Diagram and Matrix Diagram Operations

In [26], [29], and [30], a compact data structure named a *Multi-valued Decision Diagram* (MDD) is utilized for the storage of a set of states. Herein, we redefine MDDs with respect to the representation of Petri nets, and propose an efficient approach to verify current-state opacity of discrete event systems.

Let us first define the next-state function of Petri nets. We use $\hat{T}$ to denote a subset of transitions (i.e., $\hat{T} \subseteq T$) in accordance with this function, which means that the partial transitions in which we are interested may fire and yield a set of markings.

Definition 1 Given a Petri net system $\langle N, M_0 \rangle$, a marking $M \in R(N, M_0)$, and a set of transitions $\hat{T} \subseteq T$, we define the next-state function $\mathcal{N} : R(N, M_0) \times 2^T \to 2^{R(N, M_0)}$ as

$$\mathcal{N}(M, \hat{T}) = \{M' \in \mathbb{N}^m | \exists t \in \hat{T} : M[t\rangle M'\}.$$

Furthermore, given a set of transitions $\hat{T} \subseteq T$ and a set of markings $\mathcal{M} \subseteq R(N, M_0)$, we extend the next-state function to the following form

$$\mathcal{N}(\mathcal{M}, \hat{T}) = \bigcup_{M \in \mathcal{M}} \mathcal{N}(M, \hat{T}).$$

◇

In fact, the next-state function represents a "macroscopical" transition relation of a Petri net, i.e., the relation of two sets of states (pre- and post-states) decided by a set of transitions. Assume that $\mathcal{M}^0 = \{M_0\}$ represents the set of the initial marking in a Petri net and $\mathcal{M}^n$ is the marking set reachable from $\mathcal{M}^0$ in n steps, i.e., the next-state function is executed for n times ($n \in \mathbb{N}$). The notion $\mathcal{M}^n = \mathcal{N}(\mathcal{M}^{n-1}, \hat{T})$ means that a new set of states $\mathcal{M}^n$ can be generated by $\mathcal{M}^{n-1}$ after the firing of the partial transitions in $\hat{T}$. Typically, the reachability set of a Petri net $R(N, M_0)$ is the reflexive and transitive closure of $\mathcal{N}$, starting from $\mathcal{M}^0$, i.e.,

$$R(N, M_0) = \mathcal{M}^0 \cup \mathcal{N}(\mathcal{M}^0, T) \cup \mathcal{N}(\mathcal{M}^1, T) \cup \cdots = \mathcal{N}^*(\mathcal{M}^0, T).$$

Based on the theory of MDDs [33] and the notion of next-state function, we recall the definition of directed acyclic graphs and then formally redefine MDDs for storing a set of markings and matrix diagrams for transition relations of Petri nets.

Definition 2 A directed acyclic graph is a two-tuple $Z = (Q, E)$, where Q is a set of vertexes and $E \subseteq Q \times Q$ is a set of edges, directed from one vertex to another, such that there is no oriented annular path. Given an edge $e = (q, q') \in E$, it is directed from vertex q to q', where q and q' are said to be adjacent. We also say that e is an out-edge of q. Given a vertex $q \in Q$, $q^\bullet = \{q' \in Q | (q, q') \in E\}$ is said to be the postset of q and ${}^\bullet q = \{q' \in Q | (q', q) \in E\}$ is said to be the preset of q. If $q^\bullet = \emptyset$, q is a terminal vertex; otherwise, it is a non-terminal vertex. $\diamond$

Definition 3 A multi-valued decision diagram is a rooted directed acyclic graph and defined as a six-tuple $F = (Q, E, D, \delta, q_0, q_t)$, where Q is a finite set of vertexes, $E \subseteq Q \times Q$ is a set of edges, D is a set of labels, $\delta : E \to D$ is a labeling function that associates an edge $e \in E$ with a label $\omega \in D$, where different out-edges from a non-terminal vertex are labeled distinctively, i.e., $(\forall q, q', q'' \in Q)$ $(q' \neq q'')$ & $((q, q') \in E)$ & $((q, q'') \in E) \Rightarrow \delta((q, q')) \neq \delta((q, q''))$, $q_0 \in Q$ is the root vertex, and $q_t \in Q$ is the terminal vertex. $\diamond$

By its definition, a multi-valued decision diagram can be graphically represented in a hierarchical way: The root vertex q_0 is in the top and the terminal vertex q_t is in the bottom. Suppose that q_t is in level 0. Then the set $Q_1 = \cup_{q \in \{q_t\}} {}^\bullet q$ defines the vertexes in level 1 and the set $Q_2 = \cup_{q \in Q_1} {}^\bullet q$ defines the vertexes in level 2. Without loss of generality, let an MDD have $m + 1$ levels, where Q_i denotes the set of vertexes in level i. Then we have $Q_0 = \{q_t\}$, $Q_m = \{q_0\}$, $Q_i = \cup_{q \in Q_{i-1}} {}^\bullet q$, where $i = 1, 2, \ldots, m$. Let $Q_n = Q \setminus \{q_t\}$ be the set of non-terminal vertexes.

Fig. 1 shows an MDD $F = (Q, E, D, \delta, q_0, q_t)$, where $Q = Q_n \cup \{q_t\} = \{q_0, q_3^0, q_3^1, q_3^2, q_2^0, q_2^1, q_1^0, q_1^1\} \cup \{q_t\}$ is a set of vertexes with $Q_n = \{q_0,\ q_3^0,\ q_3^1,\ q_3^2,\ q_2^0,\ q_2^1,\ q_1^0,\ q_1^1\}$ being the set of non-terminal vertexes and q_t being the terminal vertex, q_0 is the root vertex, $E = \{e_0, e_1, \ldots, e_9\}$ is a set of edges, and $D = \{0, 1, 2\}$ is a set of labels. The labeling function δ is defined by $\delta(e_0) = 0, \delta(e_1) = 1, \delta(e_2) = 2, \delta(e_3) = 0, \delta(e_4) = 1, \delta(e_5) = 0, \delta(e_6) = 1, \delta(e_7) = 0, \delta(e_8) = 1$ and $\delta(e_9) = 0$. In Fig. 1, let $Q_n = Q_4 \cup Q_3 \cup Q_2 \cup Q_1$. Then, we have $Q_4 = \{q_0\}, Q_3 = \{q_3^0, q_3^1, q_3^2\}, Q_2 = \{q_2^0, q_2^1\}$ and $Q_1 = \{q_1^0, q_1^1\}$.

For the sake of brevity, given an edge $e = (q, q')$ with its label $\delta(e) = \omega$, q' is said to be the child vertex of q with respect to ω, denoted by $q[\omega]$, i.e., $q[\omega] = q'$. Let $q \in Q_n$ be a non-terminal vertex. Given $\omega \in D$, vertex q is said to be extensile with respect to ω if there exists the child vertex of

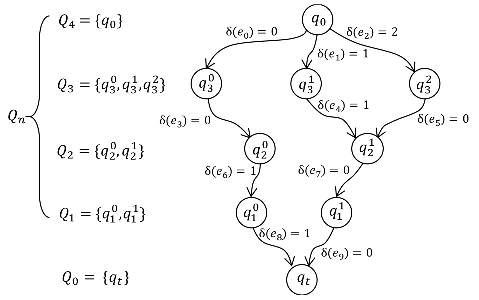


Fig. 1: An MDD.

q with respect to ω, i.e., $\delta((q, q')) = \omega$; otherwise, it is said to be not extensile with respect to ω. For example, in Fig. 1, the non-terminal vertex q_3^1 is extensile with respect to $\omega = 1$, and not extensile with respect to $\omega = 0$ or $\omega = 2$. Note that we write $q[\omega]!$ if q is extensile with respect to ω.

Definition 4 Given two MDDs $F = (Q, E, D, \delta, q_0, q_t)$ and $F' = (Q', E', D, \delta', q_0', q_t')$ with the same number of levels ($m + 1$ levels) and the same set of labels D, they are said to be isomorphic if there exists an injective and surjective function $\gamma : Q \to Q'$ such that for all $q \in Q$, if $\gamma(q) = q'$, one of the following two statements holds: (1) both q and q' are terminal vertexes, i.e., $q = q_t$ and $q' = q_t'$; (2) $(\exists i \in \{1, 2, \ldots, m\})$ $(q \in Q_i$ & $q' \in Q_i')$ & $(\forall \omega \in D)$ $(q[\omega]!$ & $q'[\omega]!) \Rightarrow \gamma(q[\omega]) = q'[\omega]$. $\diamond$

By Definition 3, an MDD contains only one root vertex and the child vertexes of any non-terminal vertex q are distinguished, i.e., $q[\omega] \neq q[\omega']$ if $q[\omega]!$, $q[\omega']!$, and $\omega \neq \omega'$. With the function γ, the root vertex q_0 in F maps the root vertex q_0' in F', the child vertex of q_0 with respect to ω, i.e., $q_0[\omega]$, maps the child vertex of q_0' with respect to ω, i.e., $q_0'[\omega]$, for all $\omega \in D$ if $q_0[\omega]!$ and $q_0'[\omega]!$, and so on all in this way down to the terminal vertex.

Definition 5 A path ζ in an MDD is a sequence of vertexes and edges starting and ending with vertexes, defined as $\zeta = q_{jm} e_{jm} q_{j(m-1)} e_{j(m-1)} \cdots q_{j1} e_{j1} q_{j0}$. If $q_{jm} = q_0$ and $q_{j0} = q_t$, ζ is said to be a top-bottom path, denoted by ζ_{tb}. The label sequence of a path in an MDD is a string consisting of the labels of the edges in the path. Formally, given a path $\zeta = q_{jm} e_{jm} q_{j(m-1)} e_{j(m-1)} \cdots q_{j1} e_{j1} q_{j0}$, its label sequence is $\varrho = \delta(e_{jm}) \delta(e_{j(m-1)}) \cdots \delta(e_{j1})$. The set of the label sequences of all top-bottom paths in an MDD F is denoted by $\mathcal{K}(F)$. $\diamond$

Definition 6 A vertex $q' \in Q$ is said to be a descendant of $q \in Q_n$ if there is a path directed from q to q'. For any non-terminal vertex $q \in Q_n$ in an MDD $F = (Q, E, D, \delta, q_0, q_t)$, the subgraph rooted by q, denoted by $F(q)$, is defined as a graph consisting of q and all of its descendants with edges and labels reserved. $\diamond$

Proposition 1 Let $F = (Q, E, D, \delta, q_0, q_t)$ be an MDD. For any non-terminal vertex $q \in Q_n$, $F(q) = (Q', E', D', \delta', q_0', q_t')$ is an MDD.

proof. The subgraph rooted by $q \in Q$ derived from F is a part of F, i.e., $Q' \subseteq Q, E' \subseteq E, D' \subseteq D, q_0' = q$ and $q_t' = q_t$, where the labels of edges in $F(q)$, i.e., $\delta'(e')$, for all edges $e' \in E'$, remain unchanged. Therefore, $F(q) = (Q', E', D', \delta', q_0', q_t')$ is an MDD. ■

Definition 7 Let q and q' be two distinct non-terminal vertexes at the same level in an MDD. They are said to be duplicated if $F(q)$ and $F(q')$ are isomorphic. Given a non-terminal vertex q in an MDD, it is said to be redundant if q is extensile with respect to all $\omega \in D$ such that $q[\omega] = q[\omega']$, for all $\omega, \omega' \in D$ with $\omega \neq \omega'$. $\diamond$

An MDD is said to be reduced if it contains no duplicated and redundant vertexes, and it is said to be quasi-reduced if it contains no duplicated vertexes. There are some reduction algorithms to transform an arbitrary MDD into a reduced or quasi-reduced version [23], [33]. For the sake of brevity, we do not discuss reduction methods for MDDs here. It has been proven that the transformation of an MDD to be quasi-reduced requires less time than a reduced one [34]. In the remainder of the paper, we assume that all MDDs have been transformed into a quasi-reduced form.

Let us now touch upon how an MDD represents a set of markings of a Petri net. Let $N = (P, T, Pre, Post)$ be a bounded Petri net with m places, where $P = \{p_1, p_2, \ldots, p_m\}$. Given an MDD $F = (Q, E, D, \delta, q_0, q_t)$ represented in a hierarchical way with $m+1$ levels, each non-terminal vertex $q_i \in Q_i$ is associated with a single place p_i in N, where $1 \leq i \leq m$. Since the Petri net is bounded, the number of tokens in place p_i, e.g., ω, can be represented by an associated vertex q_i with its one of out-edges labeled by ω ($\omega \in D$). To be specific, given an edge $e = (q_i, q_{i-1})$ ($q_i \in Q_i, q_{i-1} \in Q_{i-1}$) with a label $\delta(e) = \omega$, the number of tokens in place p_i equals to ω. Let $M = [M(p_1), M(p_2), \ldots, M(p_m)]^T$ be a marking of a Petri net. A label sequence of a top-bottom path $\varrho = \delta(e_{jm})\delta(e_{j(m-1)}) \cdots \delta(e_{j1})$ represents a marking, where $\delta(e_{jm}) = \omega_m = M(p_m), \delta(e_{j(m-1)}) = \omega_{m-1} = M(p_{m-1}), \ldots, \delta(e_{j1}) = \omega_1 = M(p_1)$.

Example 1 In Fig. 2, transitions t_1, t_3, and t_4 are observable, where t_1 and t_3 are labeled with a, and t_4 is labeled with b. Transitions t_2 and t_5 are unobservable labeled with ε. Consider a set of markings $\{[1,1,0,0]^T, [0,0,1,1]^T, [0,0,0,2]^T\}$ in LPN G. In Fig. 1, vertexes in Q_4, Q_3, Q_2 and Q_1 represent places p_4, p_3, p_2 and p_1, respectively. Three top-bottom paths $q_0 e_0 q_3^0 e_3 q_2^0 e_6 q_1^0 e_8 q_t$, $q_0 e_1 q_3^1 e_4 q_2^1 e_7 q_1^1 e_9 q_t$ and $q_0 e_2 q_3^2 e_5 q_2^1 e_7 q_1^1 e_9 q_t$ with their label sequences 0011, 1100 and 2000 represent markings $[1,1,0,0]^T$, $[0,0,1,1]^T$ and $[0,0,0,2]^T$, respectively. $\square$

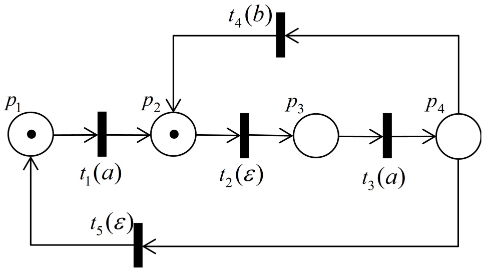
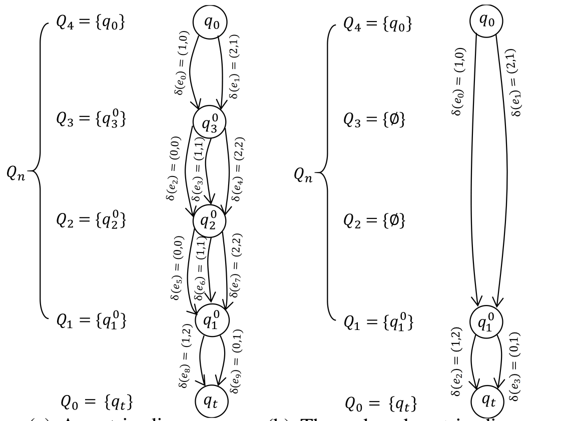


Fig. 2: An LPN with initial state $\{[1,1,0,0]^T\}$.

Transition relations in a Petri net can be represented by a matrix diagram that is similar to an MDD, except that each edge is associated with a label pair.

Definition 8 A matrix diagram is a rooted directed acyclic graph that is defined as a six-tuple $H = (Q, E, \mathcal{D}, \delta, q_0, q_t)$, where Q is a finite set of vertexes, $E \subseteq Q \times Q$ is a set of edges, $\mathcal{D} \subseteq D \times D$ is a set of label pairs, $\delta : E \to \mathcal{D}$ is a labeling function that associates an edge $e \in E$ with a label pair $(\omega_u, \omega_v) \in \mathcal{D}$, where different out-edges from a non-terminal vertex are labeled distinctively, i.e., $(\forall q, q', q'' \in Q)$ $(q' \neq q'')$ & $((q, q') \in E)$ & $((q, q'') \in E) \Rightarrow \delta((q, q')) \neq \delta((q, q''))$, $q_0 \in Q$ is the root vertex, and $q_t \in Q$ is the terminal vertex. $\diamond$

Similar to an MDD, a matrix diagram can also be graphically represented in a hierarchical way with the same manner. Fig. 3(a) shows a matrix diagram $H = (Q, E, \mathcal{D}, \delta, q_0, q_t)$, where $Q = Q_n \cup \{q_t\} = \{q_0, q_3^0, q_2^0, q_1^0\} \cup \{q_t\}$ is a set of vertexes with $Q_n = \{q_0, q_3^0, q_2^0, q_1^0\}$ being the set of non-terminal vertexes and q_t being the terminal vertex, q_0 is the root vertex, $E = \{e_0, e_1, \ldots, e_9\}$ is a set of edges, and $\mathcal{D} \subseteq D \times D$ (here $D = \{0,1,2\} \subseteq \mathbb{N}$) is a set of label pairs. The labeling function δ is defined by $\delta(e_0) = (1,0), \delta(e_1) = (2,1), \delta(e_2) = (0,0), \delta(e_3) = (1,1), \delta(e_4) = (2,2), \delta(e_5) = (0,0), \delta(e_6) = (1,1), \delta(e_7) = (2,2), \delta(e_8) = (1,2)$ and $\delta(e_9) = (0,1)$. In Fig. 3(a), let $Q_n = Q_4 \cup Q_3 \cup Q_2 \cup Q_1$. Then, we have $Q_4 = \{q_0\}, Q_3 = \{q_3^0\}, Q_2 = \{q_2^0\}$ and $Q_1 = \{q_1^0\}$.



(a) A matrix diagram. (b) The reduced matrix diagram from Fig. 3(a).

Fig. 3: Transition relations with $\hat{T} = \{t_5\}$ represented by a matrix diagram.

Definition 9 Given a non-terminal vertex q ($q \in Q_n$) in a matrix diagram, it is said to be identical if q is extensile with respect to (ω_u, ω_v) with $\omega_u = \omega_v$ and not extensile with respect to (ω_u, ω_v) with $\omega_u \neq \omega_v$, for all $(\omega_u, \omega_v) \in \mathcal{D}$. $\diamond$

A matrix diagram is said to be *reduced* if it contains no identical vertex. In what follows, all matrix diagrams are assumed to be reduced. Similar to an MDD, in a matrix diagram, each non-terminal vertex can also be associated with a place in a Petri net. Given a bounded Petri net $N = (P, T, Pre, Post)$, where $P = \{p_1, p_2, \ldots, p_m\}$, an identical vertex q_i ($q_i \in Q_i$) corresponds to the case that the tokens in place p_i ($p_i \in P$) remain unchanged after a transition fires, since the out-edges of q_i are labeled by (ω_u, ω_v) with $\omega_u = \omega_v$. Fig. 3(b) shows the reduced version of the matrix diagram in Fig. 3(a) by removing its identical vertexes (q_3^0 and q_2^0). Let $q_i \in Q_i$ be an identical vertex. If q_i is removed, its out-edges are also removed. Then, a vertex q_i' in the preset of q_i, i.e., $q_i' \in {}^\bullet q_i$, and a vertex q_i'' in the postset of q_i, i.e., $q_i'' \in q_i^\bullet$, become adjacent, implying that the identical vertex q_i is "skipped".

Definition 10 A path η in a matrix diagram is a sequence of vertexes and edges starting and ending with vertexes, defined as $\eta = q_{jm}e_{jm}q_{j(m-1)}e_{j(m-1)}\cdots q_{j1}e_{j1}q_{j0}$. If $q_{jm} = q_0$ and $q_{j0} = q_t$, η is said to be a top-bottom path, denoted by η_{tb}. The label sequence of a path in a matrix diagram is a string consisting of the labels of the edges in the path. Formally, given a path $\eta = q_{jm}e_{jm}q_{j(m-1)}e_{j(m-1)}\cdots q_{j1}e_{j1}q_{j0}$, its label sequence is $\tau = \delta(e_{jm})\delta(e_{j(m-1)})\cdots\delta(e_{j1})$. The set of the label sequences of all top-bottom paths in a matrix diagram H is denoted by $\mathcal{K}(H)$. $\diamond$

Let us now expose the mechanism of representing transition relations of a Petri net by a matrix diagram. Based on Definition 1, the next-state function is represented as $\mathcal{M}^n = \mathcal{N}(\mathcal{M}^{n-1}, \hat{T})$. In the matrix diagram, both sets $\mathcal{M}^{n-1}$ and $\mathcal{M}^n$ could be expressed. Let $N = (P, T, Pre, Post)$ be a bounded Petri net with $P = \{p_1, p_2, \ldots, p_m\}$. Given a matrix diagram $H = (Q, E, \mathcal{D}, \delta, q_0, q_t)$ shown in Fig. 3(a), represented in a hierarchical way with $m+1$ levels, each non-terminal vertex $q_i \in Q_i$ is associated with a place p_i in a Petri net. In contrast to an MDD, a non-terminal vertex q_i with an out-edge labeled by (ω_u, ω_v) represents a transition relation related to place p_i. To be specific, given an edge $e = (q_i, q_{i-1})$ ($q_i \in Q_i$, $q_{i-1} \in Q_{i-1}$) with a label pair $\delta(e) = (\omega_u, \omega_v)$, ω_u and ω_v represent the numbers of tokens in place p_i before and after the firing of a particularly related transition, respectively. A label sequence of a top-bottom path represents a state transition process, i.e., a transition fires at a marking M yielding a marking M'.

Example 2 Let us consider the LPN in Fig. 2, a set of markings $\{[1,1,0,0]^T, [0,0,1,1]^T, [0,0,0,2]^T\}$ and the next-state function with respect to $\hat{T} = \{t_5\}$. The corresponding matrix diagram is shown in Fig. 3(a). Label sequence $(1,0)(1,1)(0,0)(0,1)$ represents the state transition process $[0,0,1,1]^T[t_5\rangle[1,0,1,0]^T$, while label sequence $(2,1)(0,0)(0,0)(0,1)$ represents the state transition process $[0,0,0,2]^T[t_5\rangle[1,0,0,1]^T$. Note that if transition t_5 fires, the numbers of tokens in place p_2 and p_3 remain unchanged, and thus in the reduced matrix diagram (Fig. 3(b)), vertexes q_2^0 and q_3^0 are identical and "skipped". The next-state function in this example is written as $\{[1,0,1,0]^T, [1,0,0,1]^T\} = \mathcal{N}(\{[1,1,0,0]^T, [0,0,1,1]^T, [0,0,0,2]^T\}, \{t_5\})$. $\square$

Definition 11 Let $F' = (Q', E', D, \delta', q_0', q_t')$ and $F'' = (Q'', E'', D, \delta'', q_0'', q_t'')$ be two MDDs with the same number of levels ($m+1$ levels) and the same set of labels D. The union of F' and F'' (denoted as $F' \cup F''$) is defined as $F_u = (Q, E, D, \delta, (q_0', q_0''), (q_t', q_t''))$, where $Q = Q_m \cup Q_{m-1} \cup \cdots \cup Q_0$ and $E = E_m \cup E_{m-1} \cup \cdots \cup E_1$. $Q_m = \{(q_0', q_0'')\}$,

$$Q_{i-1} = \begin{cases} \cup\{(q_i'[\omega], q_i''[\omega])\} & \text{if } (\forall\omega \in D)\ (q_i', q_i'') \in Q_i\ \& \\ & \quad q_i'[\omega]!\ \&\ q_i''[\omega]!; \\ \cup\{q_i'[\omega]\} & \text{if } (\forall\omega \in D)\ (((q_i', q_i'') \in Q_i\ \& \\ & \quad \neg q_i''[\omega]!)\ |^{1}\ q_i' \in Q_i)\ \&\ q_i'[\omega]!; \\ \cup\{q_i''[\omega]\} & \text{if } (\forall\omega \in D)\ (((q_i', q_i'') \in Q_i\ \& \\ & \quad \neg q_i'[\omega]!)\ |\ q_i'' \in Q_i)\ \&\ q_i''[\omega]!; \\ \emptyset & \text{otherwise,} \end{cases}$$

and $E_i \subseteq Q_i \times Q_{i-1}$ ($i = m, m-1, \ldots, 1$, and if $i = 1$, $Q_{i-1} = \{(q_t', q_t'')\}$), where the labeling function with respect to e ($e \in E_i$) is defined as $\delta(e) = \omega$ (if $i = m$, $(q_m'[\omega], q_m''[\omega]) = (q_0'[\omega], q_0''[\omega])$).

The intersection of F' and F'' (denoted as $F' \cap F''$) is defined as $F_i = (Q, E, D, \delta, (q_0', q_0''), (q_t', q_t''))$, where $Q = Q_m \cup Q_{m-1} \cup \cdots \cup Q_0$ and $E = E_m \cup E_{m-1} \cup \cdots \cup E_1$. $Q_m = \{(q_0', q_0'')\}$,

$$Q_{i-1} = \begin{cases} \cup\{(q_i'[\omega], q_i''[\omega])\} & \text{if } (\forall\omega \in D)\ q_i'[\omega]!\ \&\ q_i''[\omega]!; \\ \emptyset & \text{otherwise,} \end{cases}$$

$E_i \subseteq Q_i \times Q_{i-1}$ ($i = m, m-1, \ldots, 1$, and if $i = 1$, $Q_{i-1} = \{(q_t', q_t'')\}$), where the labeling function with respect to e ($e \in E_i$) is defined as $\delta(e) = \omega$, and the vertexes $q \in Q_i$ satisfying $q^\bullet = \emptyset$ are deleted (if $i = m$, $(q_m'[\omega], q_m''[\omega]) = (q_0'[\omega], q_0''[\omega])$). $\diamond$

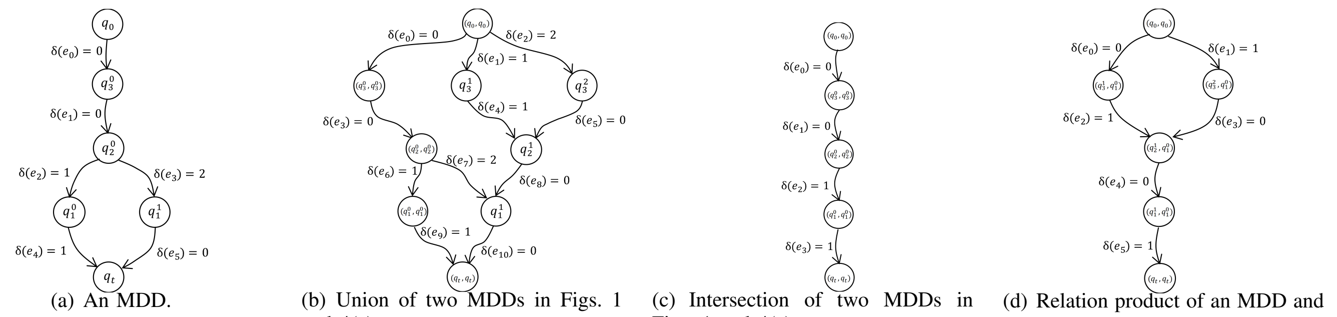
In Definition 11, for both F_u and F_i, their non-terminal vertexes are defined recursively from Q_m to Q_1. For the union of two MDDs F_u, let $q_i \in Q_i$ ($i = m, m-1, \ldots, 1$) be a non-terminal vertex. If $q_i = (q_i', q_i'')$ and for all $\omega \in D$, both q_i' and q_i'' are extensile with respect to ω, i.e., $q_i'[\omega]!$ and $q_i''[\omega]!$, then $\{q_{i-1} = (q_i'[\omega], q_i''[\omega])\} \subseteq Q_{i-1}$ is defined. If one of the two following statements holds: (1) $q_i = (q_i', q_i'')$, $q_i'[\omega]!$ and $\neg q_i''[\omega]!$, for all $\omega \in D$; (2) $q_i = q_i'$ and $q_i'[\omega]!$, for all $\omega \in D$, then $\{q_{i-1} = q_i'[\omega]\} \subseteq Q_{i-1}$ is defined.

Example 3 We consider the LPN in Fig. 2. Let $\mathcal{M}_1 = \{[1,1,0,0]^T, [0,0,1,1]^T, [0,0,0,2]^T\}$ and $\mathcal{M}_2 = \{[1,1,0,0]^T, [0,2,0,0]^T\}$ be two sets of markings that are represented by MDDs F_1 and F_2 shown in Figs. 1 and 4(a), respectively. The union and intersection of $\mathcal{M}_1$ and $\mathcal{M}_2$ are $\mathcal{M}_u = \mathcal{M}_1 \cup \mathcal{M}_2 = \{[1,1,0,0]^T, [0,0,1,1]^T, [0,0,0,2]^T, [0,2,0,0]^T\}$ and $\mathcal{M}_i = \mathcal{M}_1 \cap \mathcal{M}_2 = \{[1,1,0,0]^T\}$, respectively. By Definition 11, the union and intersection of F_1 and F_2 are shown in Figs. 4(b) and 4(c) with two sets of label sequences of top-bottom paths $\{0011, 1100, 2000, 0020\}$ and $\{0011\}$, respectively. $\square$

Proposition 2 Given two MDDs F' and F'' with two sets of label sequences $\mathcal{K}(F')$ and $\mathcal{K}(F'')$, the union (resp., intersection) of F' and F'' is an MDD F_u (resp., F_i) with $\mathcal{K}(F_u) = \mathcal{K}(F') \cup \mathcal{K}(F'')$ (resp., $\mathcal{K}(F_i) = \mathcal{K}(F') \cap \mathcal{K}(F'')$).

Proof. Let $F' = (Q', E', D, \delta', q_0', q_t')$ and $F'' = (Q'', E'', D, \delta'', q_0'', q_t'')$ be two MDDs with the same number of levels. By Definition 11, the union of F' and F'', i.e., $F_u = (Q, E, D, \delta, (q_0', q_0''), (q_t', q_t''))$, is a directed acyclic graph (with $m+1$ levels) that contains a set of vertexes Q with $Q_m = \{(q_0', q_0'')\}$ and $Q_0 = \{(q_t', q_t'')\}$, and a set of edges E, where each edge $e \in E$ is associated with a label $\omega \in D$ with $\delta(e) = \omega$ and $E_i \subseteq Q_i \times Q_{i-1}$ ($i = m, m-1, \ldots, 1$). We now show that different out-edges from a vertex $q \in Q_n$ are

[1]The notation "|" represents logical disjunction "or".



(a) An MDD.

(b) Union of two MDDs in Figs. 1 and 4(a).

(c) Intersection of two MDDs in Figs. 1 and 4(a).

(d) Relation product of an MDD and a matrix diagram in Figs. 1 and 3(b).

Fig. 4: Operations of union, intersection and relation product in Example 3.

labeled distinctively. Given two non-terminal vertexes $q_i' \in Q'$, $q_i'' \in Q''$ and a label $\omega \in D$, if $q_i'[\omega]!$ and $q_i''[\omega]!$, i.e., the edges $e_i' = (q_i', q_i'[\omega]) \in E'$ and $e_i'' = (q_i'', q_i''[\omega]) \in E''$ are associated with the same label ω, then the edges e_i', e_i'' and vertexes q_i', q_i'' are "merged". If $q_i'[\omega]!$ and $\neg q_i''[\omega]!$, or $q_i''[\omega]!$ and $\neg q_i'[\omega]!$, then the edges $e_i' = (q_i', q_i'[\omega])$ and $e_i'' = (q_i'', q_i''[\omega])$ with the same label ω do not exist. Thus, F_u is an MDD.

For all $q_i' \in Q_i'$, $q_i'' \in Q_i''$ ($i = m, m-1, \ldots, 1$), and $\omega \in D$, since the edges associated with the same labels, i.e., $e_i' = (q_i', q_i'[\omega])$ and $e_i'' = (q_i'', q_i''[\omega])$ with $q_i'[\omega]!$ and $q_i''[\omega]!$, are "merged" in label sequences of top-bottom paths, and the edges associated with different labels, i.e., $e_i' = (q_i', q_i'[\omega])$ or $e_i'' = (q_i'', q_i''[\omega])$ with $q_i'[\omega]!$ and $\neg q_i''[\omega]!$, or $q_i''[\omega]!$ and $\neg q_i'[\omega]!$, belong to separate label sequences of top-bottom paths, $\mathcal{K}(F_u) = \mathcal{K}(F') \cup \mathcal{K}(F'')$ holds.

Similar to the proof of the union of two MDDs, by Definition 11, $F_i = F' \cap F''$ is a directed acyclic graph (with $m+1$ levels) that contains a set of vertexes Q with $Q_m = \{(q_0', q_0'')\}$ and $Q_0 = \{(q_t', q_t'')\}$, and a set of edges E, where each edge $e \in E$ is associated with a label $\omega \in D$ with $\delta(e) = \omega$ and $E_i \subseteq Q_i \times Q_{i-1}$ ($i = m, m-1, \ldots, 1$). Given two non-terminal vertexes $q_i' \in Q'$, $q_i'' \in Q''$, and a label $\omega \in D$, if $q_i'[\omega]!$ and $q_i''[\omega]!$, i.e., edges $e_i' = (q_i', q_i'[\omega])$ and $e_i'' = (q_i'', q_i''[\omega])$ are associated with the same label ω, then the edges e_i', e_i'' and vertexes q_i', q_i'' are "merged". Therefore, F_i is an MDD.

For all $q_i' \in Q_i'$, $q_i'' \in Q_i''$ ($i = m, m-1, \ldots, 1$), and $\omega \in D$, since the edges associated with the same labels, i.e., $e_i' = (q_i', q_i'[\omega])$ and $e_i'' = (q_i'', q_i''[\omega])$ with $q_i'[\omega]!$ and $q_i''[\omega]!$, are "merged" in label sequences of top-bottom paths, $\mathcal{K}(F_i) = \mathcal{K}(F') \cap \mathcal{K}(F'')$ holds. ■

By Definition 11 and Proposition 2, given two sets of markings $\mathcal{M}_1$ and $\mathcal{M}_2$ in an LPN, they can be represented by two MDDs F_1 and F_2, respectively. The union (resp., intersection) of $\mathcal{M}_1$ and $\mathcal{M}_2$, i.e., $\mathcal{M} = \mathcal{M}_1 \cup \mathcal{M}_2$ (resp., $\mathcal{M} = \mathcal{M}_1 \cap \mathcal{M}_2$), can be represented by the union (resp., intersection) of MDDs F_1 and F_2, i.e., an MDD $F_u = F_1 \cup F_2$ (resp., $F_i = F_1 \cap F_2$).

Definition 12 Let $F = (Q', E', D, \delta', q_0', q_t')$ and $H = (Q'', E'', \mathcal{D}, \delta'', q_0'', q_t'')$ be an MDD and a matrix diagram with the same number of levels ($m+1$ levels), respectively, where $\mathcal{D} \subseteq D \times D$. Given two label sequences $\varrho = \omega_{jm}\omega_{j(m-1)} \cdots \omega_{j1}$ and $\tau = (\omega_{jm}, \omega_{jm}')(\omega_{j(m-1)}, \omega_{j(m-1)}') \cdots (\omega_{j1}, \omega_{j1}')$ of two top-bottom paths in F and H, respectively, the relation product of ϱ and τ is defined as a label sequence $\xi = \varrho \otimes \tau = \omega_{jm}'\omega_{j(m-1)}' \cdots \omega_{j1}'$. The relation product of F and H is defined as $F_r = F \otimes H = (Q, E, D, \delta, (q_0', q_0''), (q_t', q_t''))$, where $Q = Q_m \cup Q_{m-1} \cup \cdots \cup Q_0$ and $E = E_m \cup E_{m-1} \cup \cdots \cup E_1$. $Q_m = \{(q_0', q_0'')\}$,

$$Q_{i-1} = \begin{cases} \cup\{(q_i'[\omega], q_k''[(\omega, \omega')])\} & \text{if } (\forall \omega \in D)\ (\forall (\omega, \omega') \in \mathcal{D}) \\ & \quad q_i'[\omega]!\ \&\ q_k''[(\omega, \omega')]!\ \& \\ & \quad (i = k); \\ \cup\{(q_i'[\omega], q_k'')\} & \text{if } (\forall \omega \in D)\ q_i'[\omega]!\ \&\ (i > k); \\ \emptyset & \text{otherwise,} \end{cases}$$

($i = m, m-1, \ldots, 1$ and $1 \leq k \leq m$. Note that i and k represent the levels of vertexes in F and H, respectively, and if $i = 1$, $Q_{i-1} = \{(q_t', q_t'')\}$), $E_i \subseteq Q_i \times Q_{i-1}$ ($i = m, m-1, \ldots, 1$), where the labeling function is defined as $\delta((q_i, q_{i-1})) = \omega'$ if $q_{i-1} = (q_i'[\omega], q_k''[(\omega, \omega')])$; $\delta((q_i, q_{i-1})) = \omega$ if $q_{i-1} = (q_i'[\omega], q_k'')$, and the vertexes $q \in Q_i$ satisfying $q^\bullet = \emptyset$ are deleted (if $i = m$, $(q_m'[\omega], q_k''[(\omega, \omega')]) = (q_0'[\omega], q_0''[(\omega, \omega')])$). ◇

Similar to the operations between two MDDs, the non-terminal vertexes in F_r are defined recursively. Let $F_r = F \otimes H$. Since H is a reduced matrix diagram, its identical vertexes are removed. For any $q_i = (q_i', q_k'') \in Q_i$, if for all $\omega \in D$, $(\omega, \omega') \in \mathcal{D}$, $q_i'[\omega]!$, $q_k''[(\omega, \omega')]!$ and $i = k$, then $\{q_{i-1} = (q_i'[\omega], q_k''[(\omega, \omega')])\} \subseteq Q_{i-1}$ is defined, where $\delta((q_i, q_{i-1})) = \omega'$. If $i > k$, $\{q_{i-1} = (q_i'[\omega], q_k'')\} \subseteq Q_{i-1}$ is defined, where $\delta((q_i, q_{i-1})) = \omega$.

Example 4 We consider the LPN in Fig. 2 and a set of markings $\mathcal{M} = \{[1,1,0,0]^T, [0,0,1,1]^T, [0,0,0,2]^T\}$ that is represented by an MDD F. By Definition 12, the relation product of the MDD F (portrayed in Fig. 1) and the matrix diagram H (depicted in Fig. 3(b)) is visualized in Fig. 4(d), whose set of label sequences of top-bottom paths is $\{0101, 1001\}$. □

Proposition 3 Given an MDD $F = (Q', E', D, \delta', q_0', q_t')$ and a matrix diagram $H = (Q'', E'', \mathcal{D}, \delta'', q_0'', q_t'')$, the relation product of F and H, i.e., $F_r = F \otimes H = (Q, E, D, \delta, (q_0', q_0''), (q_t', q_t''))$, is an MDD.

Proof. By Definition 12, the relation product of F and H is a directed acyclic graph (with $m+1$ levels) that contains a set of vertexes Q with $Q_m = \{(q_0', q_0'')\}$ and $Q_0 = \{(q_t', q_t'')\}$

and a set of edges E, where each edge $e \in E$ is associated with a label $\omega \in D$ with $\delta(e) = \omega$ and $E_i \subseteq Q_i \times Q_{i-1}$ ($i = m, m-1, \dots, 1$). Given two non-terminal vertexes $q'_i \in Q'$, $q''_k \in Q''$, a label $\omega \in D$, and a label pair $(\omega, \omega') \in \mathcal{D}$, if $q'_i[\omega]!$ and $q''_k[(\omega, \omega')]!$, where $i = k$, the edge $e = ((q'_i, q''_k), (q'_i[\omega], q''_k[(\omega, \omega')])) \in E$ with respect to the label $\omega' \in D$ is generated. If $q'_i[\omega]!$ and $i > k$, then the edge $e = ((q'_i, q''_k), (q'_i[\omega], q''_k)) \in E$ with respect to the label $\omega \in D$ is generated. Since the out-edges of the non-terminal vertexes in both F and H are distinctive, the relation product of F and H, i.e., the graph F_r, is an MDD. ■

By Definition 12, the relation product of two label sequences indicates the case that a marking M represented by a top-bottom label sequence, i.e., a label sequence of a top-bottom path, can yield a marking M' under the state transition process represented by a top-bottom label sequence in a matrix diagram. The relation product of an MDD and a matrix diagram calculates the output of the next-state function, i.e., $\mathcal{M}' = \mathcal{N}(\mathcal{M}, \hat{T})$, where $\mathcal{M}$ is the input set of markings of the function represented by an MDD and the transition relations are decided by a set of transitions $\hat{T}$.

To construct the next-state function, Algorithm 1 is designed to compute the union of two MDDs. In plain words, Algorithm 1, named Union-MDD(F', F''), is used to compute the union of two different sets of markings in a Petri net. After the construction of an MDD for a set of markings and a matrix diagram for transition relations, Algorithm 2, named Relation-product(F, H), is proposed to calculate the next-state function of a Petri net.

The complexity of both Algorithms 1 and 2 is $\mathcal{O}(\sum_{i=1}^{m} |Q'_i| \times |Q''_i|)$, where $|\cdot_i|$ represents the number of vertexes in an MDD or a matrix diagram in level i.

Note that the "next-state function" slightly differs from the "transition relations". The output of the next-state function is a set of markings while the transition relations (represented by a matrix diagram) depict the relation of pre- and post-states. To distinguish the transition relations from the next-state function $\mathcal{N}$, we use $\bar{\mathcal{N}}$ to denote a matrix diagram (in Algorithms 4 and 5).

Algorithm 1: Union of two MDDs

```
Input: Two MDDs $F' = (Q', E', D, \delta', q'_0, q'_t)$ and
       $F'' = (Q'', E'', D, \delta'', q''_0, q''_t)$.
Output: An MDD $F_u = (Q, E, D, \delta, q_0, q_t)$ that is the
        union of $F'$ and $F''$.
1  $q_0 \leftarrow (q'_0, q''_0)$; $Q_m \leftarrow \{q_0\}$; $Q \leftarrow Q_m$; $E \leftarrow \emptyset$;
2  for $i = m$ to 1 do
3      Select a vertex $q_i \in Q_i$;
4      for all $\omega \in D$, s.t., $q'_i[\omega]!$ or $q''_i[\omega]!$ do
5          if $q_i = (q'_i, q''_i)$ then
6              if $q'_i[\omega]!$ and $\neg q''_i[\omega]!$ then
7                  $q_i[\omega] \leftarrow q'_i[\omega]$;
8              else if $q''_i[\omega]!$ and $\neg q'_i[\omega]!$ then
9                  $q_i[\omega] \leftarrow q''_i[\omega]$;
10             else
11                 $q_i[\omega] \leftarrow (q'_i[\omega], q''_i[\omega])$;
12         else if $q_i = q'_i$ then
13             $q_i[\omega] \leftarrow q'_i[\omega]$;
14         else
15             $q_i[\omega] \leftarrow q''_i[\omega]$;
16         if $q_i[\omega] \notin Q_{i-1}$ then
17             $Q_{i-1} \leftarrow Q_{i-1} \cup \{q_i[\omega]\}$;
18             $E \leftarrow E \cup \{e = (q_i, q_i[\omega])\}$ with its label
               $\delta(e) = \omega$;
19     $Q_i \leftarrow Q_i \setminus \{q_i\}$;
20     if $Q_i \neq \emptyset$ then
21         Go to line 3;
22     $Q \leftarrow Q \cup Q_{i-1}$;
23 $q_t \leftarrow (q'_t, q''_t) \leftarrow q'_t \leftarrow q''_t$;
```

IV. Current-state Opacity

This research focuses on the verification of current-state opacity of discrete event systems modeled with Petri nets. Here, a secret is defined as an arbitrary subset of reachable markings. In a Petri net system, the intruder has a complete knowledge of the net system $\langle N, M_0 \rangle$ with unobservable transitions. The current-state opacity of a system is formally defined as follows.

Definition 13 [11] An LPN $G = (N, M_0, \Sigma, l)$ is said to be current-state opaque with respect to a secret $S \subseteq R(N, M_0)$ if for all observations $w \in \mathcal{L}(N, M_0)$, $\mathcal{C}(w) \nsubseteq S$ holds. ◇

In plain words, if an LPN is current-state opaque, for all possible observations, the intruder is not able to decide whether the current state belongs to the secret.

A. Verifying Current-state Opacity

According to Definition 13, the verification of current-state opacity requires to check if $\mathcal{C}(w) \nsubseteq S$ is true for all $w \in \mathcal{L}(N, M_0)$, implying that the computation of $\mathcal{C}(w)$ for all observations is mandatory. Traditionally, it requires to exhaustively enumerate all sequences of transitions that might fire. Given a bounded LPN G, the key point of verifying its current-state opacity is to transfer its reachability graph to a deterministic finite automaton. Such a deterministic finite automaton is called an observer since it records all the possible sets of markings after an arbitrary observation w. However, the complexity of transferring a reachability graph to a deterministic finite automaton is $\mathcal{O}(2^{|Y|})$ [5], [35], [36], where $|Y|$ is the number of states in the reachability graph. If the reachability graph is too large, one cannot construct an observer.

To sum up, the construction of an observer requires to compute the reachability graph first, and then transfer it to a deterministic finite automaton by a standard determinization procedure [37]. The work in [11] touches upon in detail how to verify current-state opacity of a Petri net based on the observer.

Definition 14 [11] Let $G = (N, M_0, \Sigma, l)$ be an LPN and

Algorithm 2: Computation of next-state function of a Petri net

```
Input: An MDD $F = (Q', E', D, \delta', q_0', q_t')$ and a
       matrix diagram $H = (Q'', E'', \mathcal{D}, \delta'', q_0'', q_t'')$.
Output: An MDD $F_r = (Q, E, D, \delta, q_0, q_t)$ that
        represents the relation product of $F$ and $H$.
1  $q_0 \leftarrow (q_0', q_0'')$; $Q_m \leftarrow \{q_0\}$; $Q \leftarrow Q_m$; $E \leftarrow \emptyset$;
2  for $i = m$ to 1 do
3      Select a vertex $q_i = (q_i', q_k'') \in Q_i$;
4      if There exists no $\omega \in D$, s.t., $q_i'[\omega]!$ and
       $q_k''[(\omega, \omega')]!$ then
5          Go to line 18;
6      for all $\omega \in D$, s.t., $q_i'[\omega]!$ and $q_k''[(\omega, \omega')]!$ do
7          if $i = k$ then
8              for all $\omega' \in D$ s.t. $q_k''[(\omega, \omega')]!$ do
9                  $q_i[\omega'] \leftarrow (q_i'[\omega], q_k''[(\omega, \omega')])$;
10                 if $q_i[\omega'] \notin Q_{i-1}$ then
11                     $Q_{i-1} \leftarrow Q_{i-1} \cup \{q_i[\omega']\}$;
12                     $E \leftarrow E \cup \{e = (q_i, q_i[\omega'])\}$ with its
                       label $\delta(e) = \omega'$;
13         else
14             $q_i[\omega] \leftarrow (q_i'[\omega], q_k'')$;
15             if $q_i[\omega] \notin Q_{i-1}$ then
16                 $Q_{i-1} \leftarrow Q_{i-1} \cup \{q_i[\omega]\}$;
17                 $E \leftarrow E \cup \{e = (q_i, q_i[\omega])\}$ with its
                   label $\delta(e) = \omega$;
18     $Q_i \leftarrow Q_i \setminus \{q_i\}$;
19     if $Q_i \neq \emptyset$ then
20         Go to line 3;
21     $Q \leftarrow Q \cup Q_{i-1}$;
22 $q_t \leftarrow (q_t', q_t'')$;
23 Delete vertexes $q \in Q_n$ satisfying $q^\bullet = \emptyset$;
```

$S \subseteq R(N, M_0)$ be a secret. A reachable marking M is said to be exposable if it does not belong to the secret, i.e., $M \in ex(S)$, where $ex(S) = R(N, M_0) \backslash S$ is a set of exposable markings. ◇

From Definitions 13 and 14, the following fact follows.

Fact 1 An LPN G is current-state opaque with respect to S iff for all $w \in \mathcal{L}(N, M_0)$, $\mathcal{C}(w) \cap ex(S) \neq \emptyset$ holds.

Example 5 Consider the LPN in Fig. 2 and its reachability graph in Fig. 5. Given a secret $S = \{M_0, M_3, M_4, M_5\}$, the set of exposable markings is $ex(S) = \{M_1, M_2, M_6, M_7, M_8, M_9\}$. The observer of the reachability graph is shown in Fig. 6. The LPN is current-state opaque with respect to S since for all $w \in \mathcal{L}(N, M_0)$, $\mathcal{C}(w) \cap ex(S) \neq \emptyset$ holds. □

B. Multi-valued Decision Diagram for Current-state Opacity

Considering that an MDD can represent a set of markings of a Petri net and transition relations effectively, an efficient MDD-based technique is proposed to verify current-state opacity of a Petri net. Let $\mathcal{M}_1$, $\mathcal{M}_2 \subseteq R(N, M_0)$ be

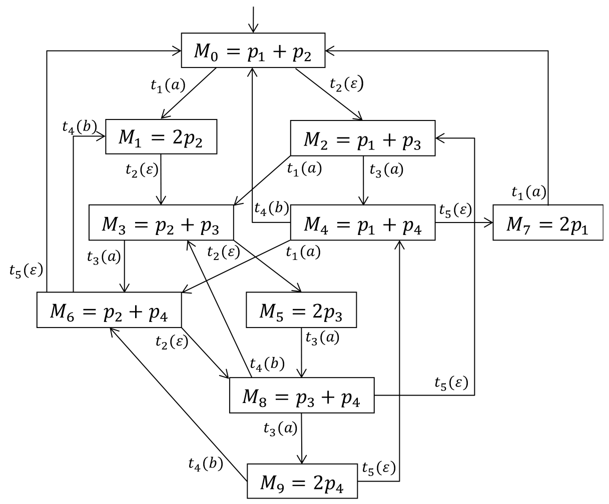


Fig. 5: The reachability graph of the LPN in Fig. 2.

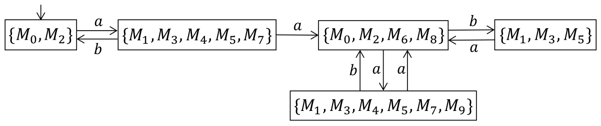

$\{M_0, M_2\}$
$\{M_1, M_3, M_4, M_5, M_7\}$
$\{M_0, M_2, M_6, M_8\}$
$\{M_1, M_3, M_5\}$
$\{M_1, M_3, M_4, M_5, M_7, M_9\}$
a
b

Fig. 6: The observer derived from the reachability graph in Fig. 5.

two sets of markings in an LPN G. Based on Definition 3, $\mathcal{M}_1$ and $\mathcal{M}_2$ can be represented by two MDDs that are denoted as F_1 and F_2, respectively. By Definition 11, we design Algorithm 3 (named Intersection-MDD(F_1, F_2)) to verify whether $\mathcal{M}_1 \cap \mathcal{M}_2 \neq \emptyset$ holds when two sets of markings are represented by MDDs.

Let $G = (N, M_0, \Sigma, l)$ be a bounded LPN. An MDD-based observer of G is represented by an MDD $F_o = (Q, E, \Sigma \cup \{\varepsilon\}, \delta, q_0, q_t)$, where each non-terminal vertex $q \in Q_n$ represents a set of markings that is structured by an MDD and $\Sigma \cup \{\varepsilon\}$ is a set of labels. To construct an MDD-based observer for a labeled Petri net, we initially define the unobservable reach and the α-reach of a set of markings $\mathcal{M}$. Then, we design an algorithm to construct an MDD-based observer.

Definition 15 Given an LPN $G = (N, M_0, \Sigma, l)$ and a set of markings $\mathcal{M} \subseteq R(N, M_0)$, the unobservable reach of $\mathcal{M}$ is defined as

$$\mathcal{N}(\mathcal{M}) = \bigcup_{M \in \mathcal{M}} \{M' \in \mathbb{N}^m | \exists \sigma_u \in T_u^* : M[\sigma_u\rangle M'\},$$

where σ_u is a sequence of unobservable transitions. It represents the unobservable reachable states generated by a set of markings $\mathcal{M}$. ◇

Definition 16 Given an LPN $G = (N, M_0, \Sigma, l)$, a label $\alpha \in \Sigma$ and a set of markings $\mathcal{M} \subseteq R(N, M_0)$, the α-reach of $\mathcal{M}$ is defined as

$$\mathcal{N}(\mathcal{M}, \alpha) = \bigcup_{M \in \mathcal{M}} \{M' \in \mathbb{N}^m | (\exists t \in T)\ l(t) = \alpha \text{ and } M[t\rangle M'\}.$$

It represents that a possible set of transitions labeled with α may fire and generate a new set of markings from $\mathcal{M}$. ◇

Algorithm 3: Intersection of two MDDs

```
Input: Two MDDs $F_1 = (Q', E', D, \delta', q'_0, q'_t)$ and
       $F_2 = (Q'', E'', D, \delta'', q''_0, q''_t)$.
Output: An MDD $F_i = (Q, E, D, \delta, q_0, q_t)$ that is the
        intersection of $F_1$ and $F_2$.
1  $q_0 \leftarrow (q'_0, q''_0)$; $Q_m \leftarrow \{q_0\}$; $Q \leftarrow Q_m$; $E \leftarrow \emptyset$;
2  for $i = m$ to 1 do
3      Select a vertex $q_i = (q'_i, q''_i) \in Q_i$;
4      if There exists no $\omega \in D$, s.t., $q'_i[\omega]!$ and $q''_i[\omega]!$
       then
5          Go to line 11;
6      for all $\omega \in D$, s.t., $q'_i[\omega]!$ and $q''_i[\omega]!$ do
7          $q_i[\omega] \leftarrow (q'_i[\omega], q''_i[\omega])$;
8          if $q_i[\omega] \notin Q_{i-1}$ then
9              $Q_{i-1} \leftarrow Q_{i-1} \cup \{q_i[\omega]\}$;
10             $E \leftarrow E \cup \{e = (q_i, q_i[\omega])\}$ with its label
               $\delta(e) = \omega$;
11     $Q_i \leftarrow Q_i \setminus \{q_i\}$;
12     if $Q_i \neq \emptyset$ then
13         Go to line 3;
14     $Q \leftarrow Q \cup Q_{i-1}$;
15 $q_t \leftarrow (q'_t, q''_t)$;
16 Delete vertexes $q \in Q_n$ satisfying $q^\bullet = \emptyset$;
```

In Definition 13, the computation of $\mathcal{C}(w)$ is critical to the verification of current-state opacity. The following theorem describes the relation of $\mathcal{N}(\mathcal{M})$, $\mathcal{N}(\mathcal{M}, \alpha)$, and $\mathcal{C}(w)$.

Theorem 1 Let $G = (N, M_0, \Sigma, l)$ be an LPN, and $w \in \mathcal{L}(N, M_0)$ be an arbitrary observation with $w = \alpha_1\alpha_2\cdots\alpha_n$ ($n \in \mathbb{N}$), where $\alpha_1, \alpha_2, \ldots, \alpha_n \in \Sigma$. $\mathcal{C}(w) = \mathcal{M}_{n+1}$ iff

$$\begin{aligned}\mathcal{M}_1 =& \mathcal{N}(\{M_0\}), \mathcal{M}_2 = \mathcal{N}(\mathcal{N}(\mathcal{M}_1, \alpha_1)), \ldots,\\ \mathcal{M}_{n+1} =& \mathcal{N}(\mathcal{N}(\mathcal{M}_n, \alpha_n)).\end{aligned}$$

proof. We use mathematical induction to prove Theorem 1. (If) Given an initial marking M_0, $\mathcal{M}_1 = \mathcal{N}(\{M_0\})$ is the set of markings due to all possible sequences of unobservable transitions from $\{M_0\}$. (1) Base case: when $n = 1$, i.e., $w = \alpha_1$, $\mathcal{N}(\mathcal{M}_1, \alpha_1)$ represents a set of markings by firing possible transitions labeled with α_1, and $\mathcal{M}_2 = \mathcal{N}(\mathcal{N}(\mathcal{M}_1, \alpha_1))$ is a set of all possible markings generated by observable transitions labeled with α_1 interleaved with unobservable transitions, i.e., $\mathcal{C}(w) = \mathcal{M}_2$. (2) Inductive step: assume that $n = k$ holds with $w = \alpha_1\alpha_2\cdots\alpha_k$. If $\mathcal{M}_1 = \mathcal{N}(\{M_0\}), \mathcal{M}_2 = \mathcal{N}(\mathcal{N}(\mathcal{M}_1, \alpha_1)), \ldots, \mathcal{M}_{k+1} = \mathcal{N}(\mathcal{N}(\mathcal{M}_k, \alpha_k))$, then $\mathcal{C}(w) = \mathcal{M}_{k+1}$ holds. When $n = k+1$, i.e., $w = \alpha_1\alpha_2\cdots\alpha_{k+1}$, $\mathcal{M}_{k+2} = \mathcal{N}(\mathcal{N}(\mathcal{M}_{k+1}, \alpha_{k+1}))$ implies that the set of markings $\mathcal{M}_{k+1}$ can generate a set of markings by the transitions labeled by α_{k+1} interleaved with unobservable transitions. Then $\mathcal{C}(w) = \mathcal{M}_{k+2}$ holds.

(Only If) Let $\mathcal{C}(w) = \mathcal{M}_{n+1}$. (1) Base case: when $n = 1$, i.e., $w = \alpha_1$ and $\mathcal{C}(w) = \mathcal{M}_2$, a set of markings can be generated by transitions labeled by α_1 interleaved with unobservable transitions. Thus, $\mathcal{M}_1 = \mathcal{N}(\{M_0\})$ and $\mathcal{M}_2 = \mathcal{N}(\mathcal{N}(\mathcal{M}_1, \alpha_1))$ hold. (2) Inductive step: assume that $n = k$ holds, i.e., $w = \alpha_1\alpha_2\cdots\alpha_k$. If $\mathcal{C}(w) = \mathcal{M}_{k+1}$, then $\mathcal{M}_1 = \mathcal{N}(\{M_0\}), \mathcal{M}_2 = \mathcal{N}(\mathcal{N}(\mathcal{M}_1, \alpha_1)), \ldots, \mathcal{M}_{k+1} = \mathcal{N}(\mathcal{N}(\mathcal{M}_k, \alpha_k))$ hold. When $n = k + 1$, i.e., $w = \alpha_1\alpha_2\cdots\alpha_{k+1}$, $\mathcal{C}(w) = \mathcal{M}_{k+2}$ implies that a set of markings can be generated from $\mathcal{C}(\alpha_1\alpha_2\cdots\alpha_k)$ when transitions labeled by α_{k+1} interleaved with unobservable transitions fire. Then $\mathcal{M}_1 = \mathcal{N}(\{M_0\}), \mathcal{M}_2 = \mathcal{N}(\mathcal{N}(\mathcal{M}_1, \alpha_1)), \ldots, \mathcal{M}_{k+2} = \mathcal{N}(\mathcal{N}(\mathcal{M}_{k+1}, \alpha_{k+1}))$ hold. ■

Assume that the set of markings $\mathcal{M}$ is expressed by an MDD $\hat{\mathcal{M}}$[2]. Then based on Algorithms 1 and 2, Algorithms 4 and 5 describe how to compute $\mathcal{N}(\hat{\mathcal{M}})$ and $\mathcal{N}(\hat{\mathcal{M}}, \alpha)$, respectively, when a set of markings is represented by an MDD $\hat{\mathcal{M}}$.

Algorithm 4: Computation of the unobservable reach of $\hat{\mathcal{M}}$

```
Input: An MDD $\hat{\mathcal{M}}$ and a matrix diagram $\bar{\mathcal{N}}_\varepsilon$ that is
       decided by unobservable transitions.
Output: A set of markings (represented by an MDD)
        generated by the firing of all possible
        unobservable transition sequences.
1 result$\leftarrow \hat{\mathcal{M}}$;
2 repeat
3     temp $\leftarrow$ result;
4     $R \leftarrow$Relation-product(temp,$\bar{\mathcal{N}}_\varepsilon$);
5     result $\leftarrow$ Union-MDD(temp,$R$);
6 until temp = result;
7 Return result.
```

Algorithm 5: Computation of the α-reach of $\hat{\mathcal{M}}$

```
Input: An MDD $\hat{\mathcal{M}}$ and a matrix diagram $\bar{\mathcal{N}}_\alpha$ that is
       decided by observable transitions labeled with
       $\alpha$.
Output: A set of markings (represented by an MDD)
        generated by the firing of all possible
        transitions labeled with $\alpha$.
1 result$\leftarrow \emptyset$;
2 $R \leftarrow$Relation-product($\hat{\mathcal{M}}, \bar{\mathcal{N}}_\alpha$);
3 result $\leftarrow R$;
4 Return result.
```

Let $\hat{\mathcal{M}}$ be a set of markings represented by an MDD. If $\mathcal{M} = \emptyset$, the MDD $\hat{\mathcal{M}}$ is said to be empty, denoted as $\hat{\mathcal{M}}\natural$. Based on Definitions 15 and 16, Algorithms 4 and 5, and Theorem 1, Algorithm 6 constructs an MDD-based observer.

Let us explain how Algorithm 6 works. The root vertex of the MDD-based observer, i.e., q_0, is initialized by the unobservable reach of $\{\hat{M}_0\}$ represented by an MDD, i,e, $\mathcal{N}(\{\hat{M}_0\})$, and the sets of vertexes Q and edges E are initialized by $\{q_0\}$ and $\emptyset$, respectively. For any vertex q in Q that has not been visited, i.e., with no tag, and for any label $\alpha \in \Sigma$, we compute the α-reach of q, i.e., $\mathcal{N}(q, \alpha)$. Then,

[2]For the sake of brevity, in the remainder of this paper, the notation "ˆ" over a notation denoting a set of markings implies that the set is represented by an MDD. Here $\hat{\mathcal{M}}$ implies that the set of markings $\mathcal{M}$ is represented by an MDD.

Algorithm 6: Construction of MDD-based observer F_o

```
Input: A bounded LPN $G = (N, M_0, \Sigma, l)$.
Output: An MDD-based observer of $G$, i.e.,
        $F_o = (Q, E, \Sigma \cup \{\varepsilon\}, \delta, q_0, q_t)$.
1  $q_0 \leftarrow \mathcal{N}(\{\hat{M}_0\})$ and assign no tag to $q_0$;
2  $Q \leftarrow \{q_0\}$; $E \leftarrow \emptyset$;
3  while non-terminal vertexes with no tag exist do
4      Select a vertex $q \in Q$ with no tag;
5      for all $\alpha \in \Sigma$ do
6          $q' \leftarrow \mathcal{N}(\mathcal{N}(q, \alpha))$;
7          if $q' \notin Q$ and $\neg q' \nmid$ then
8              $Q \leftarrow Q \cup \{q'\}$;
9              $E \leftarrow E \cup \{e = (q, q')\}$ with $\delta(e) = \alpha$;
10             Assign no tag to $q'$;
11     Tag $q$ "old";
12 Select a path $\zeta = q_{jm}e_{jm}q_{j(m-1)}e_{j(m-1)} \cdots q_{j2}e_{j2}q_{j1}$
     that contains the maximum number of vertexes and
     let $q_{j1} \in Q_1$;
13 for all $q \in Q$ s.t. $q^\bullet = \emptyset$ do
14     repeat
15         $q' \leftarrow q$;
16         $Q \leftarrow Q \cup \{q'\}$;
17         $E \leftarrow E \cup \{e = (q, q')\}$ with $\delta(e) = \varepsilon$;
18         $q \leftarrow q'$;
19     until $q \in Q_1$;
20 $Q \leftarrow Q \cup \{q_t\}$;
21 for all $q \in Q_1$ do
22     $E \leftarrow E \cup \{e = (q, q_t)\}$ with $\delta(e) = \varepsilon$;
```

calculate its unobservable reach, i.e., $\mathcal{N}(\mathcal{N}(q, \alpha))$, and assign it to a non-terminal vertex q'. If q' is not included in Q and the set of markings represented by q' is not empty, q' is added to Q as a child vertex of q, and an edge $e = (q, q')$ labeled with α is added to E. This computation repeats iteratively until no unchecked non-terminal vertex in Q is found.

Then, we select a "longest" path $\zeta = q_{jm}e_{jm}q_{j(m-1)} e_{j(m-1)} \cdots q_{j1}$ that contains the maximum number of vertexes and let $q_{j1} \in Q_1$. For all non-terminal vertexes q that have no child vertexes, we iteratively assign themselves to their child vertexes q' with edges $e = (q, q') \in E$ labeled with ε until they are included in Q_1. Finally, the terminal vertex q_t is added to Q and for all vertexes $q \in Q_1$, edges $e = (q, q_t) \in E$ labeled with ε are added to E.

With Algorithm 6, we can construct an MDD, where each non-terminal vertex is a set of markings, represented by an MDD. Therefore, the MDD-based observer is a "geminate MDD" structure. For the verification of current-state opacity with the MDD-based observer, we introduce how to use an MDD to represent a set of exposable markings, and then design an MDD-based verifier to verify whether a system is current-state opaque with respect to a secret.

Definition 17 Given an MDD-based observer $F_o = (Q, E, \Sigma \cup \{\varepsilon\}, \delta, q_0, q_t)$ with $m + 1$ levels and a set of exposable markings represented by an MDD $\hat{ex(S)}$, the exposable marking constructor with respect to $\hat{ex(S)}$ is defined as an MDD $F_e = (Q, E, \{\varepsilon\}, \delta, q_0, q_t)$ with the same number of levels as F_o ($m + 1$ levels), where $Q = \{q_0, q_{m-1}, q_{m-2}, \ldots, q_1, q_t\}$, $E = \{(q_0, q_{m-1}), (q_{m-1}, q_{m-2}), \ldots, (q_1, q_t)\}$ with $\delta(e) = \varepsilon$ for all $e \in E$, and each non-terminal vertex $q \in Q_n$ represents the set of exposable markings $\hat{ex(S)}$. ◇

We design Algorithm 7 to construct the exposable marking constructor F_e. Let G be an LPN in Fig. 2 and $\hat{ex(S)}$ be a set of exposable markings. Based on Algorithms 6 and 7, Figs. 7(a) and 7(b) present the construction procedures of an MDD-based observer of G and an exposable marking constructor with respect to $\hat{ex(S)}$, respectively.

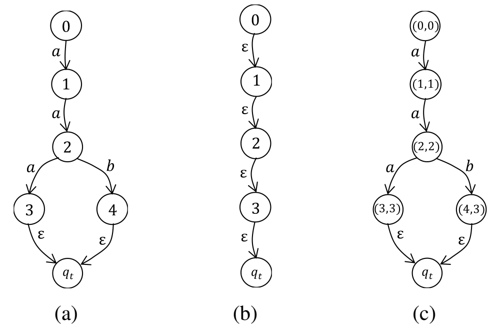


Fig. 7: (a) The MDD-based observer of the LPN G in Fig. 2, (b) the exposable marking constructor with respect to $\hat{ex(S)}$, and (c) the verifier generated by Figs. 7(a) and 7(b).

Algorithm 7: Construction of the exposable marking constructor F_e

```
Input: An MDD-based observer $F_o$ with $m + 1$ levels
       and a set of exposable markings represented by
       an MDD $\hat{ex(S)}$.
Output: An MDD for the exposable marking
        constructor $F_e$.
1  $q_0 \leftarrow \hat{ex(S)}$; $Q \leftarrow \{q_0\}$; $E \leftarrow \emptyset$;
2  $q_m \leftarrow q_0$;
3  for $i = m$ to 2 do
4      $q_{i-1} \leftarrow q_i[\varepsilon]$;
5      $Q \leftarrow Q \cup \{q_{i-1}\}$;
6      $E \leftarrow E \cup \{e = (q_i, q_{i-1})\}$ with $\delta(e) = \varepsilon$;
7  $Q \leftarrow Q \cup \{q_t\}$;
8  $E \leftarrow E \cup \{e = (q_1, q_t)\}$ with $\delta(e) = \varepsilon$;
```

Definition 18 Given an MDD-based observer $F_o = (Q_o, E_o, \Sigma \cup \{\varepsilon\}, \delta_o, q_{0o}, q_{to})$ with $m + 1$ levels and an exposable marking constructor $F_e = (Q_e, E_e, \{\varepsilon\}, \delta_e, q_{0e}, q_{te})$ with the same number of levels ($m+1$ levels), the MDD-based verifier is defined as $F_v = (Q, E, \Sigma \cup \{\varepsilon\}, \delta, q_0, q_t)$, where F_v and F_o are isomorphic, $Q_i = Q_{oi} \times Q_{ei}$ ($i = 1, 2, \ldots, m$), and each non-terminal vertex $q_i \in Q_n$ represents the intersection of two sets of markings represented by two MDDs, i.e., $q_{oi} \in Q_{oi}$ and $q_{ei} \in Q_{ei}$. ◇

Algorithm 8 is designed to construct the MDD-based verifier based on an MDD-based observer and an exposable marking constructor. Fig. 7(c) shows the verifier generated by the MDD-based observer (in Fig. 7(a)) and the exposable marking constructor (in Fig. 7(b)).

Algorithm 8: Construction of MDD-based verifier F_v

```
Input: An MDD-based observer
       $F_o = (Q_o, E_o, \Sigma \cup \{\varepsilon\}, \delta_o, q_{0o}, q_{to})$ and an
       exposable marking constructor
       $F_e = (Q_e, E_e, \{\varepsilon\}, \delta_e, q_{0e}, q_{te})$ with the same
       number of levels.
Output: An MDD-based verifier
       $F_v = (Q, E, \Sigma \cup \{\varepsilon\}, \delta, q_0, q_t)$.
1  $q_0 \leftarrow (q_{0o}, q_{0e}) \leftarrow$ Intersection-MDD($q_{0o}, q_{0e}$);
2  $Q_m \leftarrow \{q_0\}$; $Q \leftarrow Q_m$; $E \leftarrow \emptyset$;
3  for $i = m$ to 1 do
4      Select a vertex $q_i = (q_{oi}, q_{ei}) \in Q_i$;
5      for all $\alpha \in \Sigma \cup \{\varepsilon\}$ s.t. $q_{oi}[\alpha]!$ do
6          $q_i[\alpha] \leftarrow (q_{oi}[\alpha], q_{ei}[\varepsilon]) \leftarrow$
             Intersection-MDD($q_{oi}[\alpha], q_{ei}[\varepsilon]$);
7          if $q_i[\alpha] \notin Q_{i-1}$ then
8              $Q_{i-1} \leftarrow Q_{i-1} \cup \{q_i[\alpha]\}$;
9              $E \leftarrow E \cup \{e = (q_i, q_i[\alpha])\}$ with its label
                 $\delta(e) = \alpha$;
10     $Q_i \leftarrow Q_i \setminus \{q_i\}$;
11     if $Q_i \neq \emptyset$ then
12         Go to line 4;
13     $Q \leftarrow Q \cup Q_{i-1}$;
14 $q_t \leftarrow (q_{to}, q_{eo})$;
```

Theorem 2 Given an LPN $G = (N, M_0, \Sigma, l)$, a secret S, and the MDD-based verifier $F_v = (Q, E, \Sigma \cup \{\varepsilon\}, \delta, q_0, q_t)$ of G, the LPN G is current-state opaque with respect to S iff for all non-terminal vertex $q \in Q_n \subseteq Q$, $\neg q\dagger$ holds.

proof. (If) Consider an arbitrary non-terminal vertex $q \in Q_n$ of the MDD-based verifier. By Definitions 14 and 18 and Algorithms 6, 7 and 8, q is an MDD that represents the intersection of $\mathcal{C}(w)$ and $ex(S)$. If $\neg q\dagger$, i.e., the set represented by the MDD q is not an empty set, it implies that for all w, $\mathcal{C}(w) \cap ex(S) \neq \emptyset$ holds. Then by Fact 1, the system is current-state opaque with respect to S.

(Only If) Assume that there exists a non-terminal vertex q, such that $q\dagger$, i.e., the set of markings represented by q is an empty set. According to Definitions 14 and 18 and Algorithms 6, 7 and 8, there is an observation w such that $\mathcal{C}(w) \cap ex(S) = \emptyset$. Then by Fact 1, the system is not current-state opaque with respect to S. ■

Although the BRG-based approach for the verification of current-state opacity has smaller space complexity than the method based on the reachability graph, it requires to compute minimal e-vectors initially and generate new basis markings one by one. Broadly, the complexity of the construction of BRG-based observer is $\mathcal{O}(2^{|N_b|})$ with $|N_b|$ being the number of basis markings. Instead of computing the reachability graph or BRG initially, our MDD-based approach can construct the MDD-based observer directly with the complexity $\mathcal{O}(|Obs_M| \times (\sum_{i=1}^{m} |Q'_i|_{max} \times |Q''_i|_{max}))^3$. Since it is not necessary to construct the BRG, the restriction in [8], [9], [11] that the unobservable subnet of an LPN is acyclic can be discarded. We can conclude that the MDD-based method is practically much more efficient and appropriate for more general Petri nets. Some numerical results are given in Section 5.

Moreover, when the secret changes, the MDD-based observer remains unchanged. We just need to assign the set of new exposable markings $\hat{ex(S)}$ to the non-terminal vertexes of the existed exposable marking constructor, and update the MDD-based verifier. In this sense, with different secrets, the structures of F_o, F_e and F_v are robust.

Example 6 Consider the LPN G in Fig. 2 with a secret $S = \{M_0, M_3, M_4, M_5\}$. The MDD-based observer F_o of G, exposable marking constructor F_e with respect to $\hat{ex(S)}$ and MDD-based verifier F_v based on F_o and F_e are illustrated in Figs. 7(a), 7(b) and 7(c), respectively. In the MDD-based verifier F_v, there are five non-terminal vertexes and the sets of markings represented by non-terminal vertexes are shown in Table 1. Since there is no non-terminal vertex q, such that $q\dagger$, by Theorem 2, the LPN is current-state opaque with respect to S. □

TABLE I: Marking sets represented by non-terminal vertexes in the verifier shown in Fig. 7(c)

non-terminal vertexes	marking sets
(0,0)	$\{M_0\}$
(1,1)	$\{M_3, M_4, M_5\}$
(2,2)	$\{M_0\}$
(3,3)	$\{M_3, M_4, M_5\}$
(4,3)	$\{M_3, M_5\}$

V. Numerical Examples

To verify the efficiency of the MDD-based method for the verification of current-state opacity, we compare our method with the approach proposed in [11] with MATLAB tool package [38]. Our approach is based on the "Meddly" library using C++ [39] and we mainly compare the computational cost for the verification of current-state opacity among reachability graphs, BRGs and MDDs.

Consider the LPN G in Fig. 2. The number of tokens in place p_2 is represented by $k \in \mathbb{N}$. With the increase of k, a series of tests with different initial markings, i.e., $M_0 = kp_2$, and different secrets $S = \{M \in \mathbb{N}^4 | M(p_1) + M(p_4) \geq k - 2\}$, can be conducted. We design two types of labeling functions for the LPN G and compare their computational cost (illustrated in Tables 2 and 3).

The notations $|R(N, M_0)|$, $|Obs_R|$, $|Obs_B|$ and $|Obs_M|$ represent the numbers of reachable markings, states of observers based on the reachability graphs, states of BRG-based observers and non-terminal vertexes of MDD-based observers, respectively, while T-R, T-B and T-MDD record the computational cost of the verification of current-state opacity based on the methods of reachability graphs, BRGs and MDDs,

[3]Here $|Obs_M|$ represents the number of non-terminal vertexes in an MDD-based observer, while $|Q'_i|_{max}$ and $|Q''_i|_{max}$ represent the maximum numbers of vertexes at level i for all the operations between two MDDs.

TABLE II: The state information and computational cost with the LPN G under labeling function $l(t_1) = l(t_3) = l(t_4) = l(t_5) = a$ and $l(t_2) = \varepsilon$

k	$\|R(N, M_0)\|$	$\|Obs_R\|$	$\|Obs_B\|$	$\|Obs_M\|$	T-R	T-B	T-MDD	R-CSO
5	57	13	13	13	0.09	5.05	0.003	Y
10	287	23	23	23	1.01	18.17	0.01	Y
15	817	33	33	33	8.53	38.72	0.03	Y
30	5457	63	63	63	292.37	149.51	0.34	Y
50	23427	103	103	103	5758.37	422.48	2.12	Y
100	176852	203	203	203	o.t.	1994.60	21.88	Y
150	585277	303	303	303	o.t.	5524.60	108.70	Y
200	1373702	403	-	403	o.t.	o.t.	356.56	Y
250	2667126	503	-	503	o.t.	o.t.	1784.97	Y
300	4590552	603	-	603	o.t.	o.t.	4120.58	Y

TABLE III: The case that the subnet of unobservable transitions is cyclic, i.e., $l(t_1) = l(t_5) = a$ and $l(t_2) = l(t_3) = l(t_4) = \varepsilon$

k	$\|R(N, M_0)\|$	$\|Obs_R\|$	$\|Obs_B\|$	$\|Obs_M\|$	T-R	T-B	T-MDD	R-CSO
5	57	6	-	6	0.04	-	0.001	Y
10	287	11	-	11	0.53	-	0.007	Y
15	817	16	-	16	4.22	-	0.02	Y
30	5457	31	-	31	160.35	-	0.14	Y
50	23427	51	-	51	3098.10	-	0.65	Y
100	176852	101	-	101	o.t.	-	9.57	Y
150	585277	151	-	151	o.t.	-	49.94	Y
200	1373702	201	-	201	o.t.	-	137.51	Y
250	2667126	251	-	251	o.t.	-	420.11	Y
300	4590552	301	-	301	o.t.	-	972.46	Y

respectively (the time is in seconds, and o.t. means that the computation is out of time, since it cannot be calculated within 5 hours). In Column 9, R-CSO displays whether a system is current-state opaque with respect to S (the notation Y implies that the system is current-state opaque with respect to S).

In Tables 2, we are interested in the computational cost for the verification of current-state opacity based on the methods of BRGs and MDDs. For both BRG- and MDD-based methods, the computational cost in Table 2 reflected in Fig. 8 increases exponentially. However, the growth of the curve generated by BRG-based method is faster. When $k > 50$, the observers of reachability graphs cannot be constructed within 5 hours, while in the case $k > 150$, the BRG-based method for verifying current-state opacity is out of time. The notation "-" in Column 4 means that the figures in those grids cannot be calculated.

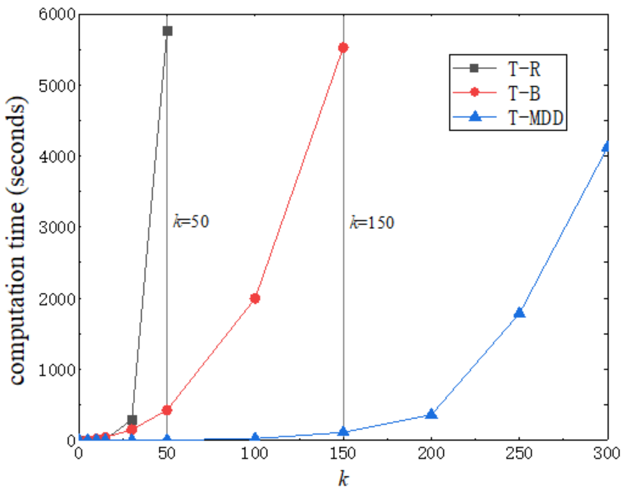


Fig. 8: Graphic representation of computing time in Table 2.

In Table 3, we consider an LPN system, where an unobservable subnet of G is cyclic. The BRGs are not able to be constructed (the notation "-" means that the parameters based on BRGs cannot be constructed). In this case, the advantages of the MDD-based method are highlighted.

We consider a parameterized manufacturing system modeled by Petri nets shown in Fig. 9 (chosen from [40]). In our experiments, a series of tests with the increase of two parameters α and β are conducted. For economy of space, we do not explain the meanings of places and transitions in the models. Given a parameterized manufacturing system modeled by LPNs, we assume that transitions $t'_{11}, t'_{12}, \ldots, t'_{1\beta}$ and $t''_{11}, t''_{12}, \ldots, t''_{1\beta}$ are labeled with a, while transitions $t'_{(\eta+1)1}, t'_{(\eta+1)2}, \ldots, t'_{(\eta+1)\beta}$ and $t''_{(\eta+1)1}, t''_{(\eta+1)2}, \ldots, t''_{(\eta+1)\beta}$ are labeled with b (the remaining transitions are unobservable labeled with ε). Let $M_0 = 2\beta p_1$ and the secret be $S = \{M \in \mathbb{N}^{1+2\times\beta\times(\eta+1)} | M(p_1) + M(p'_{21}) + M(p'_{22}) + \cdots + M(p'_{2\beta}) + M(p''_{21}) + M(p''_{22}) + \cdots + M(p''_{2\beta}) \geq 2\times\beta\}$. Table 4 shows the test results.

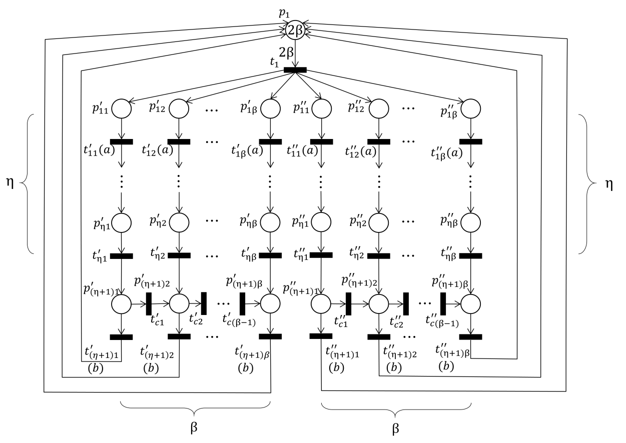


Fig. 9: A manufacturing system modeled by a parameterized Petri net.

TABLE IV: The tests on the manufacturing system in Fig. 9 with large state spaces

β	η	$\|R(N, M_0)\|$	$\|Obs_R\|$	$\|Obs_B\|$	$\|Obs_M\|$	T-R	T-B	T-MDD	R-CSO
2	3	841	14	14	23	2.95	21.13	0.03	Y
2	4	1681	14	14	23	10.69	20.96	0.04	Y
2	5	3025	14	14	23	36.67	21.25	0.06	Y
3	3	34225	27	27	52	7424.90	572.73	0.46	Y
3	4	93636	27	27	52	o.t.	575.94	1.19	Y
3	5	219961	27	27	52	o.t.	578.35	1.68	Y
4	3	1580049	44	44	93	o.t.	7684.30	12.67	Y
4	4	5885476	44	44	93	o.t.	7688.83	28.69	Y
4	5	17952169	44	44	93	o.t.	7893.60	62.71	Y
5	3	79655625	65	65	146	o.t.	o.t.	273.91	Y
5	4	403045776	65	65	146	o.t.	o.t.	803.05	Y
5	5	1592568649	65	65	146	o.t.	o.t.	1435.17	Y

In Table 4, the numbers of markings in reachability sets, i.e., $|R(N, M_0)|$, are much larger than those in the previous experiments. When $\beta \geq 3$ and $\eta \geq 4$, the method based on a reachability graph for verifying the current-state opacity of Petri nets is unpractical. When $\beta \geq 5$ and $\eta \geq 3$, the BRG-based method is not able to verify the current-state opacity of those models. It is obvious that the MDD-based method is practically efficient.

VI. Conclusion and Future Work

The main contribution made in the particular research is to excogitate a symbolic technique for verifying current-state opacity for a system modeled with labeled Petri nets. The utilization of MDDs enables us to generate markings simultaneously, and cope with the large-sized Petri nets. Also, there is no acyclic restriction on the unobservable subnet of Petri nets, since we do not need to iteratively compute the basis markings and utilize algebraic equations to generate new markings.

We plan to extend the research to the study of the other state-based opacity of LPNs such as initial-state opacity using MDDs.

References

[1] N. Busi and R. Gorrieri, "A survey on non-interference with Petri nets," in *Lectures on Concurrency and Petri Nets*. Springer, 2003, pp.328–344.

[2] M. K. Reiter and A. D. Rubin, "Crowds: Anonymity for web transactions," *ACM Trans. Inf. Syst. Secur.*, vol. 1, no. 1, pp. 66–92, 1998.

[3] Y. -C. Wu and S. Lafortune, "Comparative analysis of related notions of opacity in centralized and coordinated architectures," *Discrete Event Dyn. Syst.*, vol. 23, no. 3, pp. 307–339, 2013.

[4] A. Saboori and C. N. Hadjicostis, "Verification of initial-state opacity in security applications of discrete event systems," *Inform. Sci.*, vol. 246, pp. 115–132, 2013.

[5] F. Cassez, J. Dubreil, and H. Marchand, "Dynamic observers for the synthesis of opaque systems," in *International Symposium on Automated Technology for Verification and Analysis*. Springer, 2009, pp. 352–367.

[6] A. Saboori and C. N. Hadjicostis, "Verification of initial-state opacity in security applications of DES," in *Proc. 9th International Workshop on Discrete Event Systems*. IEEE, 2008, pp. 328–333.

[7] J. W. Bryans, M. Koutny, and P. Y. Ryan, "Modelling opacity using Petri nets," *Electronic Notes in Theoretical Computer Science*, vol. 121, pp. 101–115, 2005.

[8] Y. Tong, Z. W. Li, C. Seatzu, and A. Giua, "Verification of current-state opacity using Petri nets," in *Proc. Amer. Control Conf.*, 2015, pp. 1935–1940.

[9] Y. Tong, Z. W. Li, C. Seatzu, and A. Giua, "Verification of initial-state opacity in Petri nets," in *Proc. 54th IEEE Conf. Decision Control*, 2015, pp. 344–349.

[10] Y. Tong, Z. Y. Ma, Z. W. Li, C. Seatzu, and A. Giua, "Verification of language-based opacity in Petri nets using verifier," in *Proc. Amer. Control Conf.*, 2016, pp. 757–763.

[11] Y. Tong, Z. W. Li, C. Seatzu, and A. Giua, "Verification of state-based opacity using Petri nets," *IEEE Trans. Autom. Control*, vol. 62, no. 6, pp. 2823–2837, 2017.

[12] Y. Tong, Z. W. Li, and A. Giua, "General observation structures for Petri nets," in *Proc. 18th IEEE Conf. Emerg. Technol. Factory Automat.*, 2013, pp. 1–4.

[13] M. P. Cabasino, A. Giua, and C. Seatzu, "Fault detection for discrete event systems using Petri nets with unobservable transitions," *Automatica*, vol. 46, no. 9, pp. 1531–1539, 2010.

[14] M. P. Cabasino, A. Giua, M. Pocci, and C. Seatzu, "Discrete event diagnosis using labeled Petri nets. An application to manufacturing systems," *Control Eng. Practice*, vol. 19, no. 9, pp. 989–1001, 2011.

[15] A. Giua, C. Seatzu, and D. Corona, "Marking estimation of Petri nets with silent transitions," *IEEE Trans. Autom. Control*, vol. 52, no. 9, pp. 1695–1699, 2007.

[16] M. P. Cabasino, A. Giua, A. Paoli, and C. Seatzu, "Decentralized diagnosis of discrete-event systems using labeled Petri nets," *IEEE Trans. Syst., Man, Cybern.: Syst.*, vol. 43, no. 6, pp. 1477–1485, 2013.

[17] X. Wang, H. Hu and M. Zhou, "Discrete Event Approach to Robust Control in Automated Manufacturing Systems," *IEEE Trans. Syst., Man, Cybern.: Syst.*, doi: 10.1109/TSMC.2020.3035446.

[18] X. Wang and H. Hu, "A Robust Control Approach to Automated Manufacturing Systems Combining Absorbing and Distributing Characteristics," *IEEE Trans. Syst., Man, Cybern.: Syst.*, vol. 50, no. 12, pp. 5024-5036, 2020.

[19] C. Chen and H. Hu, "Extended Place-Invariant Control in Automated Manufacturing Systems Using Petri Nets," *IEEE Trans. Syst., Man, Cybern.: Syst.*, doi: 10.1109/TSMC.2020.3035668.

[20] Z. Y. Ma, Y. Tong, Z. W. Li, and A. Giua, "Basis marking representation of Petri net reachability spaces and its application to the reachability problem," *IEEE Trans. Autom. Control*, vol. 62, no. 3, pp. 1078–1093, 2017.

[21] H. Yue, K. Xing, H. Hu, W. Wu and H. Su, "Supervisory Control of Deadlock-Prone Production Systems with Routing Flexibility and Unreliable Resources," *IEEE Trans. Syst., Man, Cybern.: Syst.*, vol. 50, no. 10, pp. 3528-3540, 2020.

[22] A. G. Karatkevich and R. Wiśniewski, "A Polynomial-Time Algorithm to Obtain State Machine Cover of Live and Safe Petri Nets," *IEEE Trans. Syst., Man, Cybern.: Syst.*, vol. 50, no. 10, pp. 3592-3597, 2020.

[23] R. E. Bryant, "Graph-based algorithms for Boolean function manipulation," *IEEE Trans. Comput.*, vol. 100, no. 8, pp. 677–691, 1986.

[24] R. E. Bryant, "Symbolic Boolean manipulation with ordered binary-decision diagrams," *ACM Comput. Surveys*, vol. 24, no. 3, pp. 293–318, 1992.

[25] E. Pastor, J. Cortadella, and O. Roig, "Symbolic analysis of bounded Petri nets," *IEEE Trans. Comput.*, vol. 50, no. 5, pp. 432–448, 2001.

[26] A. S. Miner and G. Ciardo, "Efficient reachability set generation and storage using decision diagrams," in *International Conference on Application and Theory of Petri Nets*. Springer, 1999, pp. 6–25.

[27] G. Ciardo and A. S. Miner, "A data structure for the efficient Kronecker solution of GSPNs," in *Proc. 8th Int. Workshop Petri Nets Perform. Models*. IEEE, 1999, pp. 22–31.

[28] A. S. Miner, "Efficient solution of GSPNs using canonical matrix diagrams," in *Proc. 9th Int. Workshop Petri Nets Perform. Models*. IEEE, 2001, pp. 101–110.

[29] A. S. Miner, "Implicit GSPN reachability set generation using decision diagrams," *Perform. Eval.*, vol. 56, no. 1–4, pp. 145–165, 2004.

[30] A. S. Miner, "Saturation for a general class of models," *IEEE Trans. Softw. Eng.*, vol. 32, no. 8, pp. 559–570, 2006.

[31] S. Derisavi, P. Kemper, and W. H. Sanders, "Symbolic state-space exploration and numerical analysis of state-sharing composed models," *Linear Algebra and Its Applications*, vol. 386, pp. 137–166, 2004.

[32] G. Ciardo and R. Siminiceanu, "Using edge-valued decision diagrams for symbolic generation of shortest paths," in *International Conference on Formal Methods in Computer-Aided Design*. Springer, 2002, pp. 256–273.

[33] T. Kam, T. Villa, and R. Brayton, "Multi-valued decision diagrams: theory and applications," *Multiple-Valued Logic*, 1998.

[34] G. Ciardo, G. Lüttgen, and R. Siminiceanu, "Saturation: an efficient iteration strategy for symbolic state-space generation," in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2001, pp. 328–342.

[35] F. Cassez, J. Dubreil and H. Marchand, "Synthesis of opaque systems with static and dynamic masks," *Formal Methods Syst. Design*, vol. 40, no. 1, pp. 88–115, 2012.

[36] R. Jacob, J. -J. Lesage, and J. -M. Faure, "Overview of discrete event systems opacity: Models, validation, and quantification," *Annual Reviews in Control*, vol. 41, pp. 135–146, 2016.

[37] C. G. Cassandras and S. Lafortune, *Introduction to discrete event systems*, 2nd ed. New York, USA: Springer Science & Business Media, 2009.

[38] S. Liu, Y. Tong, C. Seatzu, and A. Giua, "Petribar: A Matlab toolbox for Petri nets implementing basis reachability approaches," *IFAC-PapersOnLine*, vol. 51, no. 7, pp. 316–322, 2018.

[39] J. Babar and A. Miner, "Meddly: Multi-terminal and edge-valued decision diagram library," in *Proc. 7th International Conference on the Quantitative Evaluation of Systems*. IEEE, 2010, pp. 195–196.

[40] M. Cabasino, A. Giua, L. Marcias, and C. Seatzu, "A comparison among tools for the diagnosability of discrete event systems," *IEEE Int. Conf. Automation Sci. Eng.*, pp. 218–223, 2012.