Sensor Selection for Structural Observability in Discrete Event Systems Modeled by Petri Nets

Yu Ru, *Student Member, IEEE,* and Christoforos N. Hadjicostis, *Senior Member, IEEE*

***Abstract*—This paper studies optimal sensor selection in discrete event systems modeled by partially observed Petri nets. The goal is to place a minimum number of sensors while maintaining structural observability, i.e., the ability to uniquely determine the system state at any given time step based on sensor information up to that time step, knowledge of the system model, and an arbitrary but known initial state. The problem is important because the majority of existing control schemes for Petri nets rely on complete knowledge of the system state at any given time step. To simplify the problem, we consider two subproblems: the optimal place sensor selection (OPSS) problem and the optimal transition sensor selection (OTSS) problem. The OPSS problem is shown to be computationally hard by establishing that the corresponding decision problem is $\mathcal{NP}$-complete. For this reason, we first reduce the problem to the linear integer programming problem, which can be solved optimally using existing linear integer programming solvers (at least for small problem instances), and then propose two heuristic algorithms to approximate its solution with polynomial complexity. Simulations suggest that the two proposed heuristics run faster and can find reasonably good solutions when compared to optimal methods that are based on linear integer programming solvers. Unlike the OPSS problem, the OTSS problem is solvable with polynomial complexity.**

***Index Terms*—Discrete event systems, Petri nets, Structural observability, Sensor selection, State-based control.**

I. Introduction

A DISCRETE event system (DES) is a dynamic system that evolves in accordance with the abrupt occurrence, at possibly unknown and irregular intervals, of physical events [1], [2]. Such systems arise in a variety of contexts, ranging from manufacturing and robotics to vehicular traffic, computer systems, and communication networks. Applications that involve monitoring and controlling of such systems rely on information conveyed by various types of sensors that are available in the system. Usually it is impossible/unnecessary to place sensors everywhere because sensors may be unavailable or prohibitively expensive for certain state transitions or other tasks. Therefore, selecting a minimum number of sensors or a set of sensors of minimal cost that also meets the system design requirements is critical and often mandatory.

This work was supported in part by the National Science Foundation (USA) under NSF ITR Award 0426831. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of NSF.

Yu Ru is with the Coordinated Science Laboratory, and the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign (e-mail: yuru2@illinois.edu). Christoforos N. Hadjicostis is with the Department of Electrical and Computer Engineering, University of Cyprus, and also with the Coordinated Science Laboratory, and the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign (e-mail: chadjic@ucy.ac.cy).

Optimal sensor selection problems have been studied extensively in discrete event systems that can be modeled as finite state machines, e.g., [3]–[6]. In [3], a sequence of tests is provided to obtain a set of sensors that has minimal cost and ensures a given property (such as diagnosability). In [4], the problem of obtaining an optimal sensor configuration of minimum cardinality is shown to be computationally hard (by showing that the corresponding decision problem is $\mathcal{NP}$-complete) for several properties, including diagnosability, normality, and observability. The authors of [5] discuss the problem of sensor selection to achieve observability with minimum cost and show that polynomial time algorithms to find good approximate solutions to this problem most likely do not exist (at least under certain complexity assumptions). Minimal[1] sensor selection to fulfill a desired formal property is shown to be generally $\mathcal{NP}$-hard in [6]; for properties that have mask-monotonic behavior (e.g., (co-)observability, normality, state-observability, and diagnosability), "top-down" and "bottom-up" methods that have polynomial complexity and achieve a minimal sensor configuration are proposed in [6].

In this paper, we focus on sensor selection in DESs that can be modeled as Petri nets. Petri nets have certain advantages over finite state machines, including a high language complexity, a compact, structural, and graphical description of the state space, and the ability to synthesize in a modular way [7]. Formal properties (e.g., observability, diagnosability) are relatively easy to define in Petri nets but difficult to check, partly because there is no practical algorithm to solve the reachability problem of Petri nets [8]. There is only limited previous work on sensor selection problems when the underlying model is a Petri net. For example, in [9], observability notions based on inputs and outputs are used as criteria when optimizing the selection of sensors in interpreted Petri net models; in this case, genetic algorithms are used to approximate the optimal sensor selection, but the method only applies to bounded Petri nets and the proposed algorithm converges slowly to a suboptimal solution.

In the sensor selection problem we consider, we formulate the notion of structural observability, i.e., the ability to uniquely determine the system state at any given time step[2]

[1]A sensor configuration is minimal if it is a minimal element in a partially ordered set [6]. In general, finding a minimal sensor configuration is easier than finding a sensor configuration with the minimum number of sensors or with the minimal cost.

[2]Time steps refer to the times at which transitions in a firing sequence fire. For example, time step i refers to the time at which the i-th transition in a firing sequence fires.

based on sensor information up to that time step, knowledge of the system model, and an arbitrary but known initial state. The requirement for uniquely determining the system state at any given time step is motivated by a number of applications where complete knowledge of the system state is absolutely necessary. Examples include the following:

- In supervisory control, control policies for a large number of synthesis methods (e.g., [10]–[17]) are defined as a function from a reachable state to a control action. Implicitly, these methods require that the system state at any given time step is exactly known. There is only a limited number of supervisor synthesis methods that are based on state estimates (e.g., [18]–[21]) because such control policies are difficult to formulate in an optimal manner [19]. The problem is that algorithms based on state estimates may also be forced to prevent transition firings that lead the system from one admissible state to another admissible state. More importantly, the use of state estimates may significantly reduce the performance of the closed-loop system and, in particular, it may lead to a deadlock [22].
- In user-interface design of safety-critical systems, discrete event systems which model such interfaces must be immediately observable in order to be "good" interfaces, i.e., interfaces that can accurately represent the underlying system to the user, so that the user will not be misled or confused [23].

Structural observability requires that the current system state is determined uniquely without delay for an arbitrary but known initial state. As shown later in the paper, even if one allows a finite delay in the definition of structural observability, the requirements for the two notions (non-delayed and delayed structural observability) remain essentially the same.

After we formulate and analyze structural observability, we consider the placement of a minimum number of sensors in the system in order to enforce this property. Unlike sensor selection problems for DES modeled as finite state machines, we allow two types of sensors (in order to model both place and transition observability): place sensors indicate the number of tokens in a particular place (e.g., vision sensors), and transition sensors indicate the firing of a transition in a given subset of transitions (e.g., motion sensors). To simplify the problem and gain a better understanding for it, we consider two subproblems: the optimal place sensor selection (OPSS) problem and the optimal transition sensor selection (OTSS) problem. The paper first establishes that the OPSS problem is computationally hard by showing that the corresponding decision problem is $\mathcal{NP}$-complete (this is done by reducing, with polynomial complexity, the minimum vertex cover (MVC) problem to this decision problem). We also show that the OPSS problem can be reduced with polynomial complexity to the minimum $\{0,1\}$-integer programming (MIP) problem which can be solved optimally using existing linear integer programming solvers (at least for small problem instances). As an alternative to the linear integer programming-based approach, we also propose two approximation algorithms to approach the optimal solution. Unlike the OPSS problem, the OTSS problem can be solved efficiently in time that is polynomial in the number of places and transitions.

The contributions of the paper include the following: i) to our best knowledge, this is the first effort to systematically investigate sensor selection problems in systems modeled by *general* Petri nets; ii) we formulate the concept of structural observability which is very important for implementing many existing state-based supervisor synthesis methods; iii) we establish two novel polynomial reductions, namely, the reduction from MVC to OPSS and the reduction from OPSS to MIP; iv) we propose two approximation algorithms (namely, the "bottom-up" algorithm and the "top-down" algorithm) that run much faster and can get reasonably good solutions to the OPSS problem, and we also obtain an upper bound on the number of place sensors in the solution generated by the "bottom-up" algorithm.

In the next section, we introduce Petri net notation and define partially observed Petri nets. In Section III, we formulate the optimal sensor selection problem while in Section IV, we give existence conditions for both the OPSS and OTSS problems. In Section V, we show that the OPSS problem is computationally hard by showing that the corresponding decision problem is $\mathcal{NP}$-complete; the problem can be solved optimally by transforming it into an MIP problem as shown in Section VI, or suboptimally using the approximation algorithms we propose and analyze in Section VII. In Section VIII, the OTSS problem is shown to be solvable with polynomial complexity. In Section IX, we compare the solutions provided by these different algorithms using a flexible manufacturing cell example. Finally, conclusions are drawn in Section X.

II. Preliminaries

In this section, we review basic definitions of Petri nets [24] and partially observed Petri nets [25].

Definition 1 A *Petri net structure* is a 4-tuple $N = (P, T, F, W)$ where $P = \{p_1, p_2, ..., p_n\}$ is a finite set of n places; $T = \{t_1, t_2, ..., t_m\}$ is a finite set of m transitions; $F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs; $W : F \to \{1, 2, 3, ...\}$ is a weight function; $P \cap T = \emptyset$ and $P \cup T \neq \emptyset$.

A *marking* is a function $M : P \to \mathcal{N}_0$ that assigns to each place a nonnegative integer number of tokens, where $\mathcal{N}_0$ denotes the set of nonnegative integers; $M(p)$ denotes the number of tokens in place p. Pictorially, places are represented by circles, transitions by bars, and tokens by black dots, as shown in Fig. 1. A *Petri net* $G = \langle N, M_0 \rangle$ is a Petri net structure N with an initial marking M_0.

A transition t is said to be *enabled* at marking M if each input place p of t (i.e., each place p such that $(p, t) \in F$) is marked with at least $W(p, t)$ tokens; this is denoted by $M[t\rangle$. The firing of enabled transition t removes $W(p, t)$ tokens from each input place p and adds $W(t, p')$ tokens to each output place p' (i.e., each place p' such that $(t, p') \in F$), resulting in a marking M'; this is denoted by $M[t\rangle M'$. In this paper, we assume that at most one transition can fire at any instant. Notation $S = t_{s_1} t_{s_2} \cdots t_{s_k}$ captures a k-length *firing sequence* from marking M if $t_{s_i} \in T$ and

$M[t_{s_1}\rangle M_1[t_{s_2}\rangle M_2 \cdots [t_{s_k}\rangle M'$; this is denoted by $M[S\rangle M'$. Marking M' can also be written as

$$M' = M + D\sigma, \tag{1}$$

where (i) D is the $n \times m$ *incidence matrix* of N satisfying $D(i,j) = -W(p_i,t_j) + W(t_j,p_i)$ (if $W(p_i,t_j)$ or $W(t_j,p_i)$ is not defined for a specific place p_i and transition t_j, it is taken to be 0), and (ii) σ is the $m \times 1$ *firing vector* of S with its ith entry being the number of times transition t_i appears in S. In this paper, we assume that the Petri net is pure, i.e., it has no self-loops.

Definition 2 Transitions t_1 and t_2 are *identically behaving* if $D(:,t_1) = D(:,t_2)$, where $D(:,t)$ denotes the column of D corresponding to transition t.

If t_1 and t_2 are identically behaving, then one of them is redundant as far as state reconstruction is concerned. Therefore, since we focus on structural observability (a system property to be defined shortly, which deals with state reconstruction), we assume without loss of generality that there are no identically behaving transitions in the Petri net we study.

Definition 3 A *partially observed Petri net* Q is a 3-tuple (N, P_o, T_o), where

- $N = (P, T, F, W)$ is a Petri net structure with n places and m transitions;
- $P_o \subseteq P$, is the set of observable places with cardinality n_1 satisfying $0 \leq n_1 \leq n$;
- $T_o \subseteq T$, is the set of observable transitions.

$P_{uo} = P \backslash P_o$ denotes the set of unobservable places. Observable places can have sensors (e.g., vision sensors) that indicate the number of tokens in a particular place, but unobservable places cannot. Similarly, $T_{uo} = T \backslash T_o$ denotes the set of unobservable transitions. Observable transitions can have sensors (e.g., motion sensors) that indicate when a transition in a given subset of transitions has fired, but unobservable transitions cannot. One can always rename places to ensure that the first n_1 places are observable; therefore, we take $P_o = \{p_1, p_2, ..., p_{n_1}\}$.

Example 1 The net in Fig. 1 is a partially observed Petri net. All places except p_4 are observable and all transitions except t_5 are observable; unobservable places (or transitions) are drawn as shadowed circles (or bars). ■

Remark 1 We exclude initial state M_0 from the definition of partially observed Petri nets (introduced in [25]) because in this paper, we focus on structural properties that are independent of M_0. ■

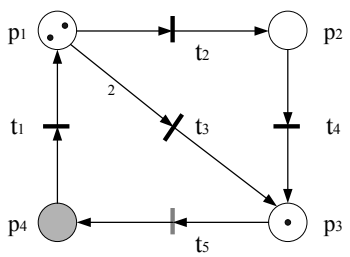


Fig. 1. A partially observed Petri net Q.

III. Problem Formulation

In order to formulate sensor selection problems, we define the notions of place sensor configuration and labeling function. A *place sensor configuration* V is a vector $(v_1\ v_2\ ...\ v_{n_1})^T$, where $v_i = 0$ if no place sensor exists on place p_i and $v_i = 1$ otherwise. $||V|| := \sum_{i=1}^{n_1} v_i \leq n_1$ denotes the total number of place sensors in the place sensor configuration V.

A *labeling function* $L : T \to \Sigma \cup \{\varepsilon\}$ assigns a label to each transition and satisfies $L(t) = \varepsilon$ for any $t \in T_{uo}$. Here, Σ is the set of labels and ε is the empty label. We define Σ so that, for each $e \in \Sigma$ there exists $t \in T_o$ satisfying $L(t) = e$. Therefore, $|\Sigma|$ is the total number of transition sensors in use and could be zero if no transition sensor is used.

When an observable transition t with a sensor fires, the label $L(t)$ is observed. Therefore, if $L(t_1) = L(t_2)$, the firings of t_1 and t_2 cannot be distinguished solely by label observation. Furthermore, if $L(t) = \varepsilon$, then the firing of transition t is not observed at all. We denote $T_e := \{t \in T : L(t) = e\}$ for all $e \in \Sigma \cup \{\varepsilon\}$.

Example 2 For the net in Fig. 1, consider $V = (1\ 1\ 1)^T$ and L defined as $L(t_1) = a$, $L(t_2) = L(t_3) = b$, $L(t_4) = c$ and $L(t_5) = \varepsilon$. Suppose $M_0 = (2\ 0\ 1\ 0)^T$ and the firing sequence t_3t_5 occurs. Then, the system trajectory is $M_0[t_3\rangle M_1[t_5\rangle M_2$, where $M_1 = (0\ 0\ 2\ 0)^T$ and $M_2 = (0\ 0\ 1\ 1)^T$. Given V and L, the available sensor information is $(2\ 0\ 1)^T \to b \to (0\ 0\ 2)^T \to (0\ 0\ 1)^T$, where $\to$ denotes the temporal order of observations. Though no label is observed when the system evolves from M_1 to M_2, we can infer that unobservable transition t_5 has occurred from the token change in place p_3, because only the firing of t_5 can decrease the token number in p_3 by 1. ■

Given a partially observed Petri net, the general sensor selection problem consists of choosing a place sensor configuration V and a labeling function L such that $||V|| + |\Sigma|$ is minimized (or, more generally, the total cost of all sensors in use is minimized) *and* the system state can be determined uniquely based on sensing information, knowledge of the system model, and an arbitrary but known initial state.

Definition 4 Given a place sensor configuration V and a labeling function L, a partially observed Petri net Q is *structurally observable* if for an *arbitrary* but known initial state M_0 and *any* firing sequence from M_0, the system state M at any given time step can be determined uniquely based on observations from place sensors and transition sensors up to that time step.

The notion of structural observability requires that the current system state is determined uniquely without delay for an arbitrary but known initial marking. If one allows a finite delay in that definition, the notion of K-delayed structural observability for a finite nonnegative integer constant K can be stated as follows, and can be shown to be equivalent to structural observability (refer to Appendix A).

Definition 5 Given a place sensor configuration V and a labeling function L, a partially observed Petri net Q is *K-delayed structurally observable* if for an *arbitrary* but known initial state M_0 and *any* firing sequence from M_0, the system state M at any given time step $i \geq 0$ can be determined uniquely based on observations from place sensors and transition sensors no later than time step $i + K$.

Remark 2 There are several properties that are related to, but differ from the notion of structural observability as defined in this work. The closest one is the immediate observability [23], i.e., the ability to determine the current state based only on partial information about the state and either the last or next event; the main difference between these two notions is that structural observability requires that the state can be determined for an *arbitrary* but known initial state. Structural marking observability [19] is defined in a different setting and has a different meaning: more specifically, in [19] each transition has a unique label and the goal is to determine if there exists an observation sequence for any *unknown* initial marking such that the current state can be reconstructed. In [26], observability involves not only the information from place sensors but also control inputs. ■

Definition 6 Given a partially observed Petri net Q and a fixed labeling function L (or a fixed place sensor configuration V), a place sensor configuration V (or a labeling function L) is *valid* if Q is structurally observable.

To simplify the sensor selection problem and gain a better understanding for it, we consider the two subproblems defined below.

Problem 1 (Optimal Place Sensor Selection (OPSS)) Given a partially observed Petri net Q and a fixed labeling function L, find a valid place sensor configuration V_{min} such that for any other valid place sensor configuration V, $||V_{min}|| \leq ||V||$.

Problem 2 (Optimal Transition Sensor Selection (OTSS)) Given a partially observed Petri net Q and a fixed place sensor configuration V, find a valid labeling function $L_{min} : T \rightarrow \Sigma \cup \{\varepsilon\}$ such that $|\Sigma|$ is minimized.

IV. Characterization of Structural Observability

According to the state equation $M = M_0 + D\sigma$, one sufficient condition for uniquely determining the system state at any given time step is that the firing of each transition at any time step can be distinguished based on sensing information. In turn, this ensures that the firing sequence and the sequence of markings can be constructed recursively. This discussion motivates the notion of transition distinguishability.

Definition 7 Given a place sensor configuration V and a labeling function L, a partially observed Petri net Q is *transition distinguishable* if, for an arbitrary but known initial state M_0, the firing of any of its transitions at any time step can be distinguished from any other transition firing based on observations from place sensors and transition sensors up to that time step.

Remark 3 Transition distinguishability is different from event-detectability as defined in [26] in that it allows information from transition sensors (besides place sensors) to be taken into account. Another related concept is invertibility [27], i.e., the ability to reconstruct the entire event string from the observation of the output string; the main difference is that invertibility allows finite delay and depends on the initial state. ■

Proposition 1 Given a place sensor configuration V and a labeling function L, a partially observed Petri net Q is structurally observable if and only if it is transition distinguishable.

Proof: (If part) If the Petri net is transition distinguishable, then we can uniquely infer the firing sequence based on transition labels and observations from place sensors. As the initial state is known, the system state can be uniquely determined using the state equation (1). This process can be continued recursively for all time steps.

(Only if part) If the Petri net is not transition distinguishable, then there exists an initial marking M_0, some time step k, and two transitions t_1 and t_2 such that the firings of t_1 and t_2 cannot be distinguished based on sensing information. The marking M at time step k enables both t_1 and t_2, and the firings of transitions t_1 and t_2 at marking M result in different markings as there are no identically behaving transitions. In this scenario, the system state cannot be determined uniquely and the Petri net is not structurally observable. ■

Proposition 1 implies that we can focus on the study of transition distinguishability. Given a place sensor configuration V, the $||V|| \times m$ matrix D_V is constructed by keeping the rows of D that correspond to observable places with sensors. In addition, for a given labeling function L, the $||V|| \times |T_e|$ matrix D_V^e is constructed for each label $e \in \Sigma \cup \{\varepsilon\}$ by keeping the columns in D_V that correspond to transitions in T_e.

Proposition 2 Given a place sensor configuration V and a labeling function L, a partially observed Petri net Q is transition distinguishable if and only if i) for each label $e \in \Sigma$, all columns of D_V^e are pairwise different, and ii) for ε, all columns of D_V^ε are nonzero and pairwise different.

Proof: The if part follows from the fact that for any transition t, there is a unique combination of a transition label $L(t)$ and a column vector of token changes $D_V(:, t)$ that identifies the firing of the transition. Now we prove the only if part by contradiction. i) Suppose there is a label $e \in \Sigma$ and two associated transitions t_1, t_2 such that $L(t_1) = L(t_2) = e$ and $D_V(:, t_1) = D_V(:, t_2)$. As there exists a marking M under which t_1 and t_2 are both enabled, we cannot distinguish transitions t_1 and t_2 based on sensor information at marking M (we can always set the initial marking to be M); this contradicts the fact that the Petri net is transition distinguishable. ii) Suppose there is a transition t such that $L(t) = \varepsilon$ and $D_V(:, t) = \mathbf{0}_{||V||}$, where $\mathbf{0}_{||V||}$ is a $||V||$-dimensional column vector with all entries being 0. As there exists a marking M such that t is enabled, then the firing of t cannot be detected based on sensing information at marking M; also a contradiction. iii) The case when there are two transitions t_1, t_2 such that $L(t_1) = L(t_2) = \varepsilon$ and $D_V(:, t_1) = D_V(:, t_2)$

can be proved in a way similar to Case i). ■

Proposition 3 Given a place sensor configuration V and a labeling function L, the transition distinguishability of a partially observed Petri net Q can be determined with complexity $\mathcal{O}(nm^2)$.

Proof: Refer to Appendix B. ■

Now we state necessary and sufficient conditions for the existence of solutions to Problems 1 and 2. First, we define $V_{max} = \mathbf{1}_{n_1}$ (where $\mathbf{1}_{n_1}$ is an n_1-dimensional column vector with all entries being 1), and define L_{max} as $L_{max}(t) = t$ for any $t \in T_o$ and $L_{max}(t) = \varepsilon$ for any $t \in T_{uo}$.

Theorem 1 (Existence Condition for OPSS) Given a partially observed Petri net Q and a fixed labeling function L, there exists an optimal place sensor configuration for the OPSS problem if and only if Q is transition distinguishable under L and V_{max}.

Proof: (If part) As the Petri net is transition distinguishable under the place sensor configuration V_{max}, it is structurally observable under V_{max} following Proposition 1, which implies that there exists at least one valid place sensor configuration. Since the total number of place sensor configurations is finite and equal to 2^{n_1}, the OPSS problem will have an optimal place sensor configuration.

(Only if part) If there exists an optimal place sensor configuration, we can add more sensors on the optimal place sensor configuration to get V_{max}, while the Petri net remains transition distinguishable following Proposition 2. ■

Example 3 Given the labeling function specified in Example 2, if we consider $V_{max} = \mathbf{1}_3$ for the net in Fig. 1, we get the 3×5 matrix below

$$D_{V_{max}} = \left|\begin{array}{c|cc|c|c} 1 & -1 & -2 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & 1 & -1 \end{array}\right| \begin{array}{c} \\ \\ \\ \end{array}$$
$$\begin{array}{ccccc} a & & b & c & \varepsilon \end{array}$$

Following Proposition 2, the Petri net is verified to be transition distinguishable, and therefore, an optimal place sensor configuration exists. This optimal place sensor configuration is found to be $V_{min} = (0\ 0\ 1)^T$ by going through all $2^3 = 8$ possible place sensor configurations. In other words, only a sensor on p_3 is needed to complement the observation of the label sequence so that we can determine the system state uniquely. ■

We also have the following condition for the OTSS problem, which can be proved in a way similar to Theorem 1.

Theorem 2 (Existence Condition for OTSS) Given a partially observed Petri net Q and a fixed place sensor configuration V, there exists an optimal labeling function for the OTSS problem if and only if Q is transition distinguishable under V and L_{max}.

Example 4 Consider the Petri net in Fig. 1 where place p_4 is unobservable and transition t_5 is unobservable. Given $V = (0\ 0\ 1)^T$, the optimal labeling function exists following Theorem 2. Intuitively, the firing of an observable transition is identifiable by its unique label specified by L_{max}, and the firing of unobservable transition t_5 is identifiable by the token change in observable place p_3. ■

Though the existence of an optimal solution to the OPSS problem can be determined with polynomial complexity, the OPSS problem itself is computationally hard (as we show in the next section, the corresponding decision problem is $\mathcal{NP}$-complete). On the other hand, as discussed in Section VIII, the OTSS problem is solvable with polynomial complexity.

V. $\mathcal{NP}$-COMPLETENESS OF OPTIMAL PLACE SENSOR SELECTION PROBLEM

We first recall some basic concepts from the field of computational complexity [28]. A problem is called a *decision problem* if all problem instances are mapped to either "true" or "false." A decision problem is said to be in the class $\mathcal{NP}$ if it can be solved by a nondeterministic Turing machine in a number of steps that is polynomial in the size of the problem. A decision problem is said to be $\mathcal{NP}$*-hard* if solving it in polynomial time would make it possible to solve all problems in the class $\mathcal{NP}$ in polynomial time. If a problem is $\mathcal{NP}$-hard and is in $\mathcal{NP}$, the problem is said to be $\mathcal{NP}$*-complete*. In this section, we show that the OPSS problem is computationally hard by demonstrating that the corresponding decision problem is $\mathcal{NP}$-complete; the $\mathcal{NP}$-hardness is established by reducing the minimum vertex cover problem (a known $\mathcal{NP}$-complete problem [28]) to the decision version of the OPSS problem.

Problem 3 (OPSS: Decision Version) Given a partially observed Petri net Q, a fixed labeling function L, and a positive integer $k \leq n$, is there a valid place sensor configuration V' such that $||V'|| \leq k$?

The definition of a vertex cover for a graph $H = (Z, E)$, where Z is the set of vertices and E is the set of edges, and the minimum vertex cover problem are recalled next.

Definition 8 Given a graph $H = (Z, E)$, a subset $Z' \subseteq Z$ is a *vertex cover* for H if for each edge $(u, v) \in E$, where $u, v \in Z$, at least one of u and v belongs to Z'.

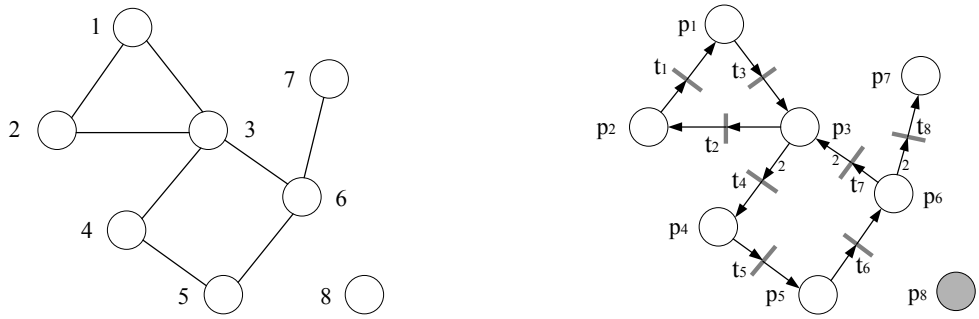
Problem 4 (Minimum Vertex Cover (MVC): Decision Version) Given a graph $H = (Z, E)$ and a positive integer $l \leq |Z|$, is there a vertex cover Z' such that $|Z'| \leq l$?

Theorem 3 The decision version of the OPSS problem is $\mathcal{NP}$-complete.

Proof: Problem 3 is shown to be $\mathcal{NP}$-complete by (i) establishing that it is in $\mathcal{NP}$ and (ii) reducing Problem 4 to it.

(i) Problem 3 is in $\mathcal{NP}$. Select a place sensor configuration V such that $||V|| \leq k$ and test if V is valid; this test can be done with complexity that is polynomial in the number of places and transitions (see Proposition 3). Therefore, Problem 3 belongs to $\mathcal{NP}$.

(ii) Problem 3 is $\mathcal{NP}$-hard because Problem 4 can be reduced to it with polynomial complexity. Given a graph $H = (Z, E)$, where $Z = \{z_1, z_2, z_3, \ldots, z_n\}$, we construct



(a) Graph (b) Corresponding net

Fig. 2. Reducing MVC to OPSS.

a partially observed Petri net Q, with n places and $m := |E|$ transitions. The vertices of the graph H are renamed to be $\{1, 2, \ldots, n\}$ so that all isolated vertices have indices $n_1 + 1, \ldots, n$. Petri net Q is constructed by making a copy of the renamed graph and attaching p_i to vertex i for $i = 1, 2, ..., n$. For every edge e with vertices i, j in H, a transition t is inserted in Q and connected to places p_i and p_j. We arbitrarily assign directions to these arcs with the constraint that the two arcs associated with the same edge in H have the same direction (see Fig. 2). Furthermore, for each vertex, we assign different weights to each incoming (or outgoing) arc; these weights range from 1 to the number of incoming (or outgoing) arcs. Places $p_1, \ldots, p_{n_1}$ (or $p_{n_1+1}, \ldots, p_n$) are taken to be observable (or unobservable). Transitions are named as t_1, t_2, ..., t_m and share the same empty label ε. Fig. 2(b) shows a Petri net constructed from the graph in Fig. 2(a) using the above mentioned procedure. The construction of the Petri net from the given graph has complexity $\mathcal{O}(|E| \times |Z|)$ that is polynomial in the number of edges and vertices. It is worth noting the following:

- The constructed partially observed Petri net Q has $P_o = \{p_1, p_2, ..., p_{n_1}\}$, $T_o = \emptyset$, and fixed labeling function $L(t_i) = \varepsilon$ for $i = 1, 2, ..., m$, which implies that all transitions are unobservable. Given this Petri net and a positive integer $k = l$, we have an instance of Problem 3. As there are no identically behaving transitions in the constructed net Q, we can decide whether a place sensor configuration V is valid or not by checking transition distinguishability following Proposition 1.
- All transitions are labeled with the same empty label, and the input place and output place of each transition are observable (unobservable places are isolated). Each observable place can be used to distinguish all of its input and output transitions as the firing of these transitions will result in different token changes in the place (recall the procedure of assigning weights to arcs). Therefore, we can distinguish a transition by putting a sensor on either its input place or output place.

Note that there is a one-to-one correspondence between a valid place sensor configuration V' for Q and a vertex cover Z' in H: (i) given any valid place sensor configuration V' for Q, each transition must have at least one input or output place p such that $V'(p) = 1$ (if $V'(p) = 0$ for both the input place and the output place of the transition, then the unobservable transition cannot be detected and therefore, the Petri net is not transition distinguishable); then, the set of vertices that correspond to the set of places that have sensors in V' is also a vertex cover for the graph H; (ii) given any vertex cover Z', for each edge $e \in E$, at least one of its two vertices v belongs to Z'; then, the transition corresponding to the edge e in H can be distinguished from other transitions by the place corresponding to the selected vertex v; therefore, we have established the validity of place sensor configuration V' (with $V'(p) = 1$ iff p corresponds to a vertex in the cover Z'). ■

VI. Transformation of OPSS to a Linear Integer Programming Problem

In this section, we convert the OPSS problem to the minimum $\{0, 1\}$-integer programming problem (a known $\mathcal{NP}$-complete problem [29]) so that it can be solved optimally using existing linear integer programming solvers (this is possible for small problem instances). Before we present the formal transformation, we first define the minimum $\{0, 1\}$-integer programming problem and then use an example to illustrate the main idea.

Problem 5 (Minimum $\{0, 1\}$-Integer Programming (MIP)) Given a $q \times s$ integer matrix A, a q-dimensional integer column vector b, and an s-dimensional nonnegative integer column vector c, find a binary s-dimensional column vector x to minimize $c^T x$ subject to $Ax \geq b$.

Example 5 For the Petri net in Fig. 1 with the labeling function defined in Example 2, we can formulate the following linear integer programming problem corresponding to the OPSS problem

$$\begin{aligned} \min \quad & c^T x \\ s.t. \quad & Ax \geq b \end{aligned}$$

where $c = (1\ 1\ 1)^T$, $x = V = (v_1\ v_2\ v_3)^T$ (as p_1, p_2 and p_3 are all observable), $b = (1\ 1)^T$, and[3]

$$A = \begin{bmatrix} -1 \neq -2 & 1 \neq 0 & 0 \neq 1 \\ 0 \neq 0 & 0 \neq 0 & -1 \neq 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}.$$

[3]Here, for integers a and b, $a \neq b$ has value 1 if a is not equal to b, and 0 otherwise.

The first row of matrix A is obtained by comparing the first three entries of $D(:,t_2)$ with the corresponding entries of $D(:,t_3)$ and captures the requirement that the place sensor configuration V distinguish transitions t_2 and t_3; the second row is obtained by comparing the first three entries of $D(:,t_5)$ with three 0's and captures the requirement that V detect unobservable transition t_5. Using the linear integer programming solver [30], the optimal solution is found to be $(0\ 0\ 1)^T$, which is the same as the solution obtained via exhaustive search in Example 3. ■

To translate an instance of the OPSS problem to an instance of the MIP problem, we define the parameters of the MIP as follows:

- Set $s = n_1$ and $q = |T_\varepsilon| + \sum_{e\in\Sigma\cup\{\varepsilon\},|T_e|\geq 2}\binom{|T_e|}{2}$, where $\binom{n}{r}$ is the binomial coefficient "n choose r";
- Set $b = \mathbf{1}_q$, $c = \mathbf{1}_s$ and $x = V$;
- Set A to be a $q \times s$ binary matrix with two kinds of rows: a) for each pair $t_j, t_k \in T_e$ $(j \neq k)$ for any label $e \in \Sigma \cup \{\varepsilon\}$ with $|T_e| \geq 2$, there is a row of the form $(D(1,j) \neq D(1,k)\ \ D(2,j) \neq D(2,k)\ \ \cdots\ \ D(n_1,j) \neq D(n_1,k))$; b) for each $t_j \in T_\varepsilon$, there is a row of the form $(D(1,j) \neq 0\ \ D(2,j) \neq 0\ \ \cdots\ \ D(n_1,j) \neq 0)$.

Proposition 4 The minimum $\{0,1\}$-integer programming problem constructed above is equivalent to the OPSS problem.

Proof: As $x = V$, $c^T x = \sum_{i=1}^{n_1} v_i = ||V||$. We need to show $Ax \geq b$ if and only if V is valid, i.e., the Petri net is structurally observable under V and the given labeling function L. Following Proposition 1 and Proposition 2, we only need to show $Ax \geq b$ if and only if all columns of D_V^e are pairwise different for each label $e \in \Sigma$ such that $|T_e| \geq 2$, and all columns of D_V^ε are nonzero and pairwise different. This is established from the following facts:

- $(D(1,j) \neq D(1,k)\ \ D(2,j) \neq D(2,k)\ \ \cdots\ \ D(n_1,j) \neq D(n_1,k))x \geq 1$ is equivalent to the fact that the jth and kth columns of incidence matrix D differ in at least one of the observable places that are equipped with a sensor (as indicated by the place sensor configuration $V = x$). The construction of matrix A ensures that this is true for all pairs $t_j, t_k \in T_e$ $(j \neq k)$ for any label $e \in \Sigma \cup \{\varepsilon\}$ with $|T_e| \geq 2$. In total, there are $\sum_{e\in\Sigma\cup\{\varepsilon\},|T_e|\geq 2}\binom{|T_e|}{2}$ inequalities of this type;
- $(D(1,j) \neq 0\ \ D(2,j) \neq 0\ \ \cdots\ \ D(n_1,j) \neq 0)x \geq 1$ is equivalent to the fact that the jth column is nonzero in at least one of the observable places that are equipped with a sensor. The construction of matrix A ensures that this is true for any $t_j \in T_\varepsilon$. In total, there are $|T_\varepsilon|$ inequalities of this type.

■

It can be verified that the reduction from OPSS to MIP can be performed with complexity that is polynomial in the number of places and transitions. Therefore, we can solve the OPSS problem optimally using linear integer programming solvers. Though linear integer programming solvers will generally be more efficient than exhaustive search, they can only effectively deal with small problem instances as the OPSS problem is computationally hard. Therefore, it is imperative to find approximation algorithms that lead to good suboptimal solutions with reasonable computational effort. This is done in the next section.

VII. Approximating Optimal Place Sensor Selection

In this section, we propose two approximation algorithms for the OPSS problem by generalizing the heuristics used for the optimal selection of diagnostic probes in [31]. More specifically, we consider how the addition of a sensor to an observable place serves to detect/distinguish unobservable transitions or to distinguish transitions that are associated with the same label $e \in \Sigma$. We then define a "measure of distinguishability" (called scoring function) and use it to facilitate our choice of observable places to put sensors on.

Given a partially observed Petri net Q with n places, m transitions, and a fixed labeling function L, if there exists one or more unobservable transitions, we construct a new partially observed Petri net $\bar{Q}$ by adding an isolated transition t_{m+1} to Q and by assigning the empty label ε to it. In other words, the new incidence matrix $\bar{D} = (D\ \mathbf{0}_n)$ if there exists at least one unobservable transition in Q; otherwise, $\bar{D} = D$. Given a place sensor configuration V, it can be shown (following Proposition 2) that if, for each label $e \in \Sigma \cup \{\varepsilon\}$, all columns of $\bar{D}_V^e$ are different from each other, then the original Petri net Q is transition distinguishable. We introduce $\bar{D}$ because i) it simplifies Proposition 2 in the sense that we only need to check if any two columns of $\bar{D}_V^e$ are different for $e \in \Sigma \cup \{\varepsilon\}$, including the empty label; and ii) it eliminates the need to treat differently unobservable transitions (whose firings generate token changes at place p) from unobservable transitions (whose firings do not generate token changes at p) in the partition of T_ε generated by observable place p (as defined below).

Definition 9 Given a partially observed Petri net Q and a fixed labeling function L, $\Omega_e(p_i)$ for $e \in \Sigma \cup \{\varepsilon\}$ denotes the *partition of T_e generated by observable place p_i* in the Petri net $\bar{Q}$, and is defined as $\Omega_e(p_i) = \{S_1, S_2, ..., S_k\}$, where

- k is equal to the number of distinct entries in the row vector $\bar{D}_V^e$, where V has $V(i) = 1$ and all other entries equal to 0;
- $S_1 \cup S_2 \cup \cdots \cup S_k = T_e$ and $S_i \cap S_j = \emptyset$ for different i, j;
- S_l for $l = 1, 2, ..., k$ is a non-empty set with the maximal number of transitions $\{t_j, ..., t_k\}$ that satisfy $t_j, \cdots, t_k \in T_e$ and $\bar{D}(i,j) = \cdots = \bar{D}(i,k)$.

Example 6 We find the partitions generated by observable place p_1 in the Petri net constructed from the net in Fig. 1 by adding an isolated transition t_6. Then the row of the incidence matrix $\bar{D}$ corresponding to place p_1 is

$$\underset{a}{|\ 1\ |}\ -1\quad \underset{b}{-2\ |}\ \underset{c}{0\ |}\ \underset{\varepsilon}{0\quad 0\ |},$$

and by definition, the partitions generated by place p_1 for labels a, b, c and ε are respectively $\{\{t_1\}\}$, $\{\{t_2\},\{t_3\}\}$, $\{\{t_4\}\}$ and $\{\{t_5, t_6\}\}$. ■

Remark 4 The partition defined above generalizes the measure of the diagnostic power of a probe in [31] in the sense that the incidence matrix $\bar{D}$ can have arbitrary integer entries while the dependency matrix in [31] only has 0 or 1 entries. ■

If there exists at least one unobservable transition in Q, then $|T_\varepsilon| \geq 2$ in $\bar{Q}$. Clearly for a label e with only one transition, the transition is observable and uniquely identified by its label. The transitions associated with a label $e \in \Sigma \cup \{\varepsilon\}$ satisfying $|T_e| \geq 2$ need to be distinguished using place sensors so that transition distinguishability holds. If the partition generated by an observable place p for such a label e has $|T_e|$ elements, then a place sensor at p is sufficient for distinguishing all transitions in T_e. In Example 6, p_1 is sufficient for distinguishing transitions associated with label b, but is insufficient for distinguishing transitions associated with ε.

Now we define the *scoring function* for an observable place p_i as $f(p_i) = \sum_{e \in \Sigma \cup \{\varepsilon\}, |T_e| \geq 2} |\Omega_e(p_i)|$. The scoring function satisfies

$$l_b \leq f(p_i) \leq u_b,$$

where $l_b = \sum_{e \in \Sigma \cup \{\varepsilon\}, |T_e| \geq 2} 1$ and $u_b = \sum_{e \in \Sigma \cup \{\varepsilon\}, |T_e| \geq 2} |T_e|$, because $1 \leq |\Omega_e(p_i)| \leq |T_e|$. If $f(p_i) > f(p_j)$, then a place sensor at p_i can distinguish more (partitions of) transitions than a place sensor at p_j; therefore, $f(p_i)$ is a measure of the ability of place p_i to distinguish transitions. If $f(p_i)$ is equal to u_b, then p_i is sufficient for distinguishing all transitions.

With the scoring functions computed, one natural "top-down" method to approximate the OPSS problem is to start with all observable places and then subtract places one by one in the order of increasing value of the scoring function, until we reach a set of observable places that cannot be reduced further without affecting the transition distinguishability of the net. The details are given in Algorithm 1.

Algorithm 1 Top-down method

Input: A partially observed Petri net Q and a fixed labeling function L

Output: V – an approximation of V_{min}

1: Determine whether there exists an optimal place sensor configuration using Theorem 1; if one does not exist, exit;
2: Construct $\bar{Q}$ by adding an isolated transition if there is at least one unobservable transition;
3: Compute $f(p_i)$ for every $p_i \in P_o$;
4: $V_{current} \Leftarrow V_{max}$ and $P_{left} \Leftarrow P_o$;
5: **while** $|P_{left}| > 0$ **do**
6: $V_{temp} \Leftarrow V_{current}$;
7: Find $p \in P_{left}$ to minimize $f(p)$. If there are multiple places that minimize $f(p)$, randomly choose one;
8: $V_{temp}(p) \Leftarrow 0$;
9: Test whether $\bar{Q}$ is transition distinguishable under V_{temp}. If it is, then $V_{current} \Leftarrow V_{temp}$;
10: $P_{left} \Leftarrow P_{left} - \{p\}$;
11: **end while**
12: $V \Leftarrow V_{current}$.

We briefly explain Lines 4-12 of Algorithm 1. After computing the score for every observable place, we set $V_{current}$ to be V_{max} and P_{left} to be the set of observable places at Line 4. P_{left} keeps track of the set of observable places that we have not considered so far and its cardinality decreases by one after each loop in the **while** clause; this guarantees the termination of the algorithm. In the loop, we use V_{temp} to represent the place sensor configuration induced from $V_{current}$ by removing a sensor from the place p that minimizes the score among all places in P_{left}. If the system is transition distinguishable under V_{temp} and L, then the sensor can be removed and we update $V_{current}$ using V_{temp}; otherwise, the sensor cannot be removed at the current iteration and $V_{current}$ remains the same as before. In both cases, we remove the corresponding place p from P_{left}. After we have considered all observable places, i.e., P_{left} is empty, the algorithm stops with the output $V_{current}$, which is an approximation to V_{min} because in any execution of the **while** loop, $V_{current}$ is always valid.

For certain Petri nets, it may be the case that some $f(p_i)$ is very close to u_b so that one or two additional place sensors are sufficient to guarantee structural observability. We next consider how to extend the scoring function to multiple places so that we can approximate the optimal solution in a "bottom-up" fashion. Suppose we have a partition $\Omega_e(p_i)$ generated by place p_i for label e and[4] $|\Omega_e(p_i)| < |T_e|$. If we choose another place p_j, we can refine the partition $\Omega_e(p_i)$ by considering whether transitions belonging to some set $S \in \Omega_e(p_i)$ can be distinguished using p_j. If we denote the refined partition as $\Omega_e(p_i, p_j)$, we can define $f(\{p_i, p_j\}) = \sum_{e \in \Sigma \cup \{\varepsilon\}, |T_e| \geq 2} |\Omega_e(p_i, p_j)|$. By repeatedly applying the above operation, we can define the scoring function $f(S)$ for any nonempty set S of observable places. Note that the partitions in $\Omega_e(p_{i_1}, p_{i_2}, ..., p_{i_k})$ are independent of the ordering of places and $f(S)$ still satisfies $l_b \leq f(S) \leq u_b$ for any nonempty set of observable places.

The idea of the "bottom-up" method is as follows: initially, choose the place p_i which maximizes $f(p_i)$ and keep its generated partition for each label; the second time around, choose the place p_j which maximizes $f(\{p_i, p_j\})$ and keep the refined partition for each label; keep doing this until $f(S) = u_b$ for some subset S of observable places. The details are given in Algorithm 2.

We briefly explain Lines 3-10 of Algorithm 2. We first set $V_{current}$ to be the place sensor configuration without sensors at Line 3. If $\bar{Q}$ is transition distinguishable under the given labeling function L and the place sensor configuration $V_{current}$, then no place sensor is necessary and the algorithm exits with $V = \mathbf{0}_{n_1}$. Otherwise we set P_{left} to be the set of observable places, S to be the empty set, and $sign$ to be 0 at Line 4. P_{left} keeps track of the set of observable places that we have not considered so far, variable S keeps track of the set of observable places that have sensors in $V_{current}$, and $sign$ has a value 1 if we have found a valid place sensor configuration

[4]If $|\Omega_e(p_i)| = |T_e|$ for every label e, then adding another place cannot increase the value of the scoring function. In fact, in such case, a sensor at place p_i is sufficient for structural observability and the proposed Algorithm 2 would terminate after selecting p_i.

Algorithm 2 Bottom-up method

Input: A partially observed Petri net Q and a fixed labeling function L

Output: V – an approximation of V_{min}

1: Determine whether there exists an optimal place sensor configuration using Theorem 1; if one does not exist, exit;
2: Construct $\bar{Q}$ by adding an isolated transition if there is at least one unobservable transition;
3: $V_{current} \Leftarrow \mathbf{0}_{n_1}$. If $\bar{Q}$ is transition distinguishable under L and $V_{current}$, exit with $V = \mathbf{0}_{n_1}$;
4: $P_{left} \Leftarrow P_o$, $S \Leftarrow \emptyset$, and $sign \Leftarrow 0$;
5: **while** $sign$ is 0 **do**
6: Find $p \in P_{left}$ to maximize $f(S \cup \{p\})$ and keep their generated partitions for each label. If there are multiple places that result in an equal maximum, randomly choose one;
7: $V_{current}(p) \Leftarrow 1$, $P_{left} \Leftarrow P_{left} - \{p\}$ and $S \Leftarrow S \cup \{p\}$;
8: If $f(S)$ is equal to u_b, then $sign \Leftarrow 1$;
9: **end while**
10: $V \Leftarrow V_{current}$.

and 0 otherwise. In the **while** loop, we first select place p (in P_{left}) that maximizes[5] the score $f(S \cup \{p\})$; then, we set $V_{current}(p)$ to be 1, remove p from P_{left}, and at the same time add p into S; finally, we determine if $f(S)$ is equal to u_b, or equivalently, if $V_{current}$ is valid. If it is, then the **while** loop ends by setting $sign$ to be 1 and $V_{current}$ is an approximation to V_{min}; otherwise, the algorithm goes to the next loop. The algorithm is guaranteed to stop because after each iteration, $|S|$ will be increased by 1 and $|S|$ is upper bounded by the number of observable places (and also because the existence of an optimal solution has been verified at Line 1).

Example 7 For the Petri net in Fig. 1 with the labeling function defined in Example 2, we illustrate the use of the "top-down" and "bottom-up" methods to solve the OPSS problem. An optimal solution exists as shown in Example 3. As t_5 is unobservable, we add an isolated transition t_6 to construct $\bar{Q}$. We first use the "top-down" method. We compute $f(p_i)$ for $i = 1, 2, 3$, and obtain $f(p_1) = 3$, $f(p_2) = 3$ and $f(p_3) = 4$. At the first iteration, p_1 and p_2 both minimize $f(p)$; we choose to eliminate the sensor on p_1 and find out that $\bar{Q}$ is still transition distinguishable. At the second iteration, we eliminate the sensor on p_2 while at the third iteration, we cannot eliminate any sensor and the algorithm ends with $V = (0\ 0\ 1)^T$. Using the "bottom-up" method, we initialize $S = \emptyset$. At the first iteration, we find p_3 which maximizes $f(p)$ and set $S = \{p_3\}$. As $f(S) = u_b = 4$, the algorithm ends with the $V = (0\ 0\ 1)^T$. For this particular example, both algorithms give the optimal solution that we obtained earlier using exhaustive search (in Example 3) and linear integer programming solvers (in Example 5). ■

[5]Instead of computing all partitions generated by $S \cup \{p\}$, we can save computation by computing the value of $f(S \cup \{p\})$ based on the value of $f(S)$ and the increment induced by place p (recall that Algorithm 2 iterates through places in a bottom-up fashion).

Now we provide a simple analysis of the performance of the "bottom-up" method by providing an upper bound on the number of place sensors in the solution generated by Algorithm 2. The upper bound is $\min(n_1,\ u_b - l_b + 1)$, where n_1 is the number of observable places and u_b (or l_b) is the maximum (or minimum) value of the scoring function. The reasoning is the following: at each iteration in Algorithm 2, one place sensor will be added and the scoring function increases at least by l_b for the first picked place and at least by 1 for the following picked places (otherwise, we will not select the place sensor); therefore, the loop will be executed at most n_1 times as there can be at most n_1 sensors to add, and also at most $u_b - l_b + 1$ times as the value of the scoring function increases from at least l_b to u_b with the increment being at least 1. For the OPSS problem in Example 7, as $n_1 = 3$, $u_b = 4$ and $l_b = 2$, we get the upper bound $\min(n_1,\ u_b - l_b + 1) = \min(3,\ 4 - 2 + 1) = 3$; clearly, the bound holds as there is only one sensor in the optimal solution.

It can be shown that both Algorithm 1 and Algorithm 2 have complexity $\mathcal{O}(n^2 m^2)$; details are omitted due to lack of space. We can obtain a better solution if we apply Algorithm 1 after Algorithm 2 by setting $V_{current}$ (at Line 4 of Algorithm 1) to be the output of Algorithm 2 instead of V_{max}; this is illustrated in Section IX-A.

VIII. Optimal Transition Sensor Selection Problem

Section V showed that the optimal *place* sensor selection problem is computationally hard by showing that the corresponding decision problem is $\mathcal{NP}$-complete. Perhaps surprisingly, the optimal *transition* sensor selection problem is solvable with complexity that is polynomial in the number of places and transitions. We first define the partition of T_o generated by a place sensor configuration V, which is a slightly modified version of Definition 9.

Definition 10 Given a partially observed Petri net Q and a fixed place sensor configuration V, the *partition of the set of observable transitions T_o generated by V* is defined to be $\Omega(V) = \{S_0, S_1, S_2, ..., S_k\}$, where

- S_0 is a (possibly empty) set with the maximal number of transitions $\{t_j, ..., t_l\}$ that satisfy $t_j, \cdots, t_l \in T_o$ and $D_V(:, j) = \cdots = D_V(:, l) = \mathbf{0}_{||V||}$ (where $D_V(:, j)$ denotes the jth column of matrix D_V);
- If S_0 is $\emptyset$, then k is equal to the number of distinct columns in the matrix D_V; if S_0 is the same as T_o, then k is defined to be 1 and $S_1 = \emptyset$; otherwise, k is equal to the number of distinct columns in the matrix D_V minus 1;
- $S_0 \cup S_1 \cup S_2 \cup \cdots \cup S_k = T_o$ and $S_i \cap S_j = \emptyset$ if $i \neq j$;
- S_i for $i = 1, 2, ..., k$ is a (possibly empty) set with the maximal number of transitions $\{t_j, ..., t_l\}$ that satisfy $t_j, \cdots, t_l \in T_o$ and $D_V(:, j) = \cdots = D_V(:, l)$ being nonzero.

Essentially, any two transitions that have the same column in D_V fall into the same partition in $\Omega(V)$. After computing the partition $\Omega(V)$, we need to configure transition sensors to distinguish/detect all transitions in S_0 by assigning a unique label to each transition in S_0 because no token changes are available for these transitions; we also need to configure transition sensors to distinguish all transitions in S_i for $i = 1, 2, ..., k$ by assigning a unique label to any $|S_i|-1$ transitions in S_i (the transition that is left without a label can be distinguished by the token changes in V). The minimum number of labels needed is equal to $|\Sigma|_{min} = \max(|S_0|, \max(|S_1|, |S_2|, \ldots, |S_k|)-1)$. If $|\Sigma| < |\Sigma|_{min}$ (i.e., if the number of available transition sensors is less than $|\Sigma|_{min}$), then by Dirichlet's drawer principle, either some transition in S_0 cannot be detected or at least two transitions in some S_l for $1 \leq l \leq k$ cannot be distinguished. Based on this idea, we have Algorithm 3 for the OTSS problem.

Algorithm 3 Algorithm for OTSS

Input: A partially observed Petri net Q and a fixed place sensor configuration V

Output: A valid labeling function $L_{min} : T \to \Sigma \cup \{\varepsilon\}$ satisfying $L_{min}(t) = \varepsilon$ for any $t \in T_{uo}$

1: Determine whether there exists an optimal labeling function using Theorem 2; if one does not exist, exit;
2: Compute the partition $\Omega(V)$ to get $S_0, S_1, S_2, \ldots, S_k$;
3: $|\Sigma| \Leftarrow \max(|S_0|, \max(|S_1|, |S_2|, \ldots, |S_k|) - 1)$. If $|\Sigma|$ is 0, exit with $L_{min}(t) = \varepsilon$ for $t \in T$; else, $\Sigma \Leftarrow \{e_1, e_2, \ldots, e_{|\Sigma|}\}$;
4: Assign a unique label from Σ to each transition in S_0 if S_0 is nonempty;
5: **for** $l = 1$ to k **do**
6: **if** $|S_l|$ is 1 **then**
7: Assign the empty label to the unique transition in S_l;
8: **else if** $|S_l| > 1$ **then**
9: Assign a unique label from Σ to each transition among any $|S_l| - 1$ transitions in S_l, and assign the empty label to the remaining transition;
10: **end if**
11: **end for**
12: $L_{min}(t) \Leftarrow \varepsilon$ for $t \in T_{uo}$ and output L_{min}.

We briefly explain Lines 3-12 of Algorithm 3. After computing $\Omega(V)$, at Line 3 we set $|\Sigma|$ (the number of transition sensors) to be $\max(|S_0|, \max(|S_1|, |S_2|, \ldots, |S_k|) - 1)$ as discussed previously. If $|\Sigma| = 0$, then no transition sensor is required; otherwise, we need $|\Sigma|$ transition labels $e_1, e_2, \ldots, e_{|\Sigma|}$. Then we define the labeling function L_{min} for observable transitions in Lines 4-11 (and for unobservable transitions at Line 12). At Line 4 we assign a unique label to each transition in S_0 if S_0 is not empty. In the **for** loop from Line 5 to Line 11, we assign labels to each transition in S_l for $l = 1, ..., k$. There are two cases: i) if $|S_l|$ is 1, then the token change can uniquely identify the only transition in S_l and therefore, we assign the empty label to this transition; ii) if $|S_l|$ is larger than 1, then we assign a unique label from $|\Sigma|$ to each transition among any $|S_l| - 1$ transitions and assign the empty label to the remaining transition. The way to assign transition labels is possible due to the value of $|\Sigma|$. L_{min} and the given V guarantee transition distinguishability and therefore, L_{min} is valid (and the number of transition labels is also minimized). Note that L_{min} is not necessarily unique as there are no constraints on how to assign labels for transitions within a set S_l or two different sets S_{l_1} and S_{l_2}.

It can be shown that Algorithm 3 has complexity $\mathcal{O}(nm^2)$; details are omitted due to lack of space.

Example 8 We consider the OTSS problem as stated in Example 4. Recall that the optimal labeling function exists if we are given place sensor configuration $V = (0\ 0\ 1)^T$. The partition of T_o generated by V is $S_0 = \{t_1, t_2\}$ and $S_1 = \{t_3, t_4\}$ by examining the matrix $D_V = (0\ 0\ 1\ 1\ -1)$. Therefore, we can set $|\Sigma| = \max(|S_0|, |S_1| - 1) = 2$ and $\Sigma = \{e_1, e_2\}$. Labeling function L defined as $L(t_1) = L(t_3) = e_1$, $L(t_2) = e_2$ and $L(t_4) = L(t_5) = \varepsilon$ is one optimal solution. ■

Remark 5 Though there can be multiple optimal labeling functions, we may form preferences depending on other criteria; for example, using the results in [32], we may want to minimize the number of possible states that are consistent with the observation of labels in case of place sensor failures. ■

In the OTSS problem, we have made one implicit assumption: a nonempty label can be associated to any subset of observable transitions. This assumption may not be realistic in certain applications due to other constraints. For example, the labeling function may be required to be injective over observable transitions, which means every observable transition should be associated with a unique (possibly empty) label. For the OTSS problem under this constraint, it can be shown (following almost the same reasoning as for the OTSS problem) that the minimum number of transition labels $|\Sigma|$ is $|S_0| + \max(|S_1| + |S_2| + \cdots + |S_k| - 1,\ 0)$ as we can assign the empty label to one of these observable transitions (which cause token changes in observable places with sensors). More specifically, $|\Sigma|$ is equal to $|T_o|$ if $S_0 = T_o$, and $|T_o| - 1$ otherwise.

IX. Example: Automated Guided Vehicles

In this section, we first consider a practical example to demonstrate the two approximation algorithms and then discuss how to modify our heuristics to solve the OPSS problem with arbitrary nonnegative integer costs.

A. Automated Guided Vehicles

In this subsection, we consider the OPSS problem in a flexible manufacturing cell modeled as a Petri net with 64 places and 53 transitions (first introduced in [10]), and use this example to compare our approximation algorithms against the method based on linear integer programming solvers. The cell includes three workstations, two part-receiving stations and one completed parts station. Five automated guided vehicles (AGV's) transport material between pairs of stations and they may collide with each other in shared regions. For more details

on the Petri net model of the cell, refer to Fig. 2 in [10] (but keep in mind that for our simulations on sensor selection, we do not need the control places in that figure). In [10], the collision avoidance problem was studied under the assumption that the current marking of the system is known. The same problem was also studied in [33] under the assumption that all transitions are observable. However, these methods cannot be directly applied when only part of the system state is known or when there are unobservable transitions.

We model the cell as a partially observed Petri net and assume that all 64 places and all 53 transitions are observable so that we do not need to worry about the existence of a solution to the OPSS problem. To test the effectiveness of our approximation methods, we need to generate labeling functions. First, we specify the number of transition labels i. Then, we let i take values $10, 13, 16, 20, 24$, and for each value of i, we randomly generate 5 labeling functions in the following manner: we allow each transition t (among all 53 transitions) to have any of the i labels with equal probability. In total, we have 25 randomly generated labeling functions; then we solve the 25 OPSS problems using Algorithm 1, Algorithm 2, and the MIP-based method (the solver we used is the open source mixed-integer programming solver [30]). Simulation programs were written in Matlab and were run on a 1.4Ghz laptop. The results obtained using the two approximation algorithms and the MIP-based method are shown in Table I. In this table, "i" refers to the number of transition labels and "# constraints" refers to the parameter q in Problem 5 of Section VI. Note that the exhaustive search method is prohibitive for this example as there are $2^{64} = 1.8447 \times 10^{19}$ possibilities.

We compare the two approximation algorithms with the MIP-based method using the difference between the number of sensors in their respective solutions, and their running time. Table II shows the comparison results when considering the difference Δ between the number of sensors given by approximation algorithms and the number given by the MIP-based method. For the "bottom-up" method, nearly one half of 25 simulations give solutions with the same number of sensors[6] as the MIP-based method and also nearly another one half give a solution that is only slightly worse (namely, the difference Δ satisfies $\Delta = 1$). For the "top-down" method, 4 of 25 simulations give solutions with the same number of sensors as the MIP-based method and over one half among the 25 simulations give a solution that is only slightly worse (namely, $\Delta = 1$). When comparing the two approximation algorithms directly, the "bottom-up" method gives better solutions for 13 simulations while the "top-down" method gives better solutions only for 1 simulation; there are 11 simulations in which the two methods give solutions with the same number of sensors. Table I shows that the running time of the "bottom-up" method is slightly longer than the running time of the "top-down" method; the running time of both approximation algorithms is much shorter than the running time of the MIP-based method, especially when there are fewer labels. For example, in the last simulation of 10 labels, the "top-down" method ran in 0.406 seconds and the "bottom-up" method ran in 2.312 seconds while the MIP-based method ran in 6287.8 seconds. These simulations suggest that the two approximation algorithms run faster and can find reasonably good solutions compared with the MIP-based method. In particular, the "bottom-up" method is quite promising in terms of running time and quality of the approximation.

[6]Though the place sensor configurations given by the "bottom-up" method and the MIP-based method have the same number of place sensors, they are not necessarily identical.

TABLE II
COMPARISON OF APPROXIMATION ALGORITHMS WITH MIP-BASED METHOD OVER 25 SIMULATIONS

Δ	Top-down Method	Bottom-up Method
0	4	12
1	14	12
2	7	1

A better solution can be achieved if we apply Algorithm 1 after Algorithm 2 by setting $V_{current}$ (at Line 4 of Algorithm 1) to be the output of Algorithm 2 instead of V_{max}. In one simulation with i being 20, both the "top-down" method and the "bottom-up" method obtain place sensor configurations with 18 sensors; however, the combined method obtains a place sensor configuration with 17 sensors, which is the same as the number of sensors obtained by the MIP-based method.

B. OPSS with Arbitrary Nonnegative Integer Costs

In this subsection, we consider a weighted version of the OPSS problem. More specifically, we associate with each observable place p_i a nonnegative integer $\mathrm{cost}(p_i)$ which captures the cost of a sensor on place p_i. Given a partially observed Petri net Q and a fixed labeling function L, we try to find a valid sensor configuration V_{min} such that for any other valid V, $C^T V_{min} \leq C^T V$, where $C = (\mathrm{cost}(p_1)\ \mathrm{cost}(p_2) \cdots \mathrm{cost}(p_{n_1}))^T$.

The existence condition for an optimal solution is still given by Theorem 1, which can be proved in a way similar to that of Theorem 1. To solve the problem, we can transform it into an integer programming problem by setting the vector c (in Problem 5 of Section VI) to be C. Notice that integer programming solvers will give the optimal solution for this problem but will be slow for large problem instances. To employ the "top-down" method developed in Section VII, we use the following modified scoring function $f'(p_i) = \frac{f(p_i)}{\mathrm{cost}(p_i)}$, where $f(p_i)$ is the scoring function defined in Section VII. The justification is the following: i) the larger the value of $f(p_i)$, the fewer place sensors are needed based on the result in Section VII; ii) the smaller the value of $\mathrm{cost}(p_i)$, the smaller the total cost. To use the "bottom-up" method, we can generalize $f'(p_i)$ to a set of places S as $f'(S) = \frac{f(S)}{\sum_{p \in S} \mathrm{cost}(p)}$. Other scoring functions with properties similar to i) and ii) above can also be used but are not studied here in the interest of space.

To compare the two modified approximation algorithms and the MIP-based method for the OPSS problem with costs, we still use the AGV example. In our simulations, we randomly generate a labeling function with 20 labels and choose 5

TABLE I
SIMULATION RESULTS OF TWO APPROXIMATION ALGORITHMS AND MIP-BASED METHOD

	Top-down Method		Bottom-up Method		MIP-based Method		
i	time (s)	# sensors	time (s)	# sensors	time (s)	# sensors	# constraints
24	0.343	18	2.079	17	1.421	16	56
	0.359	17	2.079	17	2.703	16	59
	0.359	17	1.891	16	1.094	15	47
	0.344	17	1.875	16	0.984	16	56
	0.344	16	1.969	16	2.641	16	52
20	0.375	19	2.093	18	16.391	17	67
	0.359	19	2.032	18	16.813	17	68
	0.344	18	1.953	17	5.578	17	69
	0.391	20	2.359	18	32.250	18	60
	0.375	18	2.125	18	20.875	18	69
16	0.375	20	2.297	21	266.797	19	87
	0.360	21	2.297	20	192.984	19	74
	0.359	19	2.172	19	101.391	19	85
	0.360	21	2.391	21	801.890	20	80
	0.359	21	2.266	20	1595.000	20	95
13	0.375	21	2.219	21	287.843	20	102
	0.359	22	2.250	21	1782.700	21	108
	0.391	22	2.140	22	1377.100	21	111
	0.375	21	2.250	20	630.703	20	107
	0.391	21	2.219	21	130.766	20	103
10	0.375	22	2.172	22	3899.1	22	137
	0.406	23	2.203	22	2097.5	22	162
	0.406	23	2.281	23	3575.9	22	140
	0.407	24	2.328	22	3533.2	22	130
	0.406	23	2.312	23	6287.8	22	135

cost functions: i) function 1 has entries 0 or 1 with equal probability; ii) function 2 (or 3, 4, 5) is a shifted version of function 1 by changing the expectation to be 1.5 (or 2.5, 5.5, 10.5). The results are shown in Table III. In this table, "i" refers to the ith cost function, "cost" refers to the total cost of the corresponding place sensor configuration. The results show that the two approximation algorithms give solutions close to the optimal one but with much less running time, especially when the costs of places do not exhibit large relative difference. If we fix the cost function but change the labeling function (while keeping the total number of labels to be 20), the total cost and running time for all three methods do not change much, and we omit the outcomes of these simulations.

X. CONCLUSION

This paper studies the optimal sensor selection problem to achieve structural observability in partially observed Petri nets. The OPSS problem is shown to be computationally hard, and solvable optimally via a transformation into an MIP problem or suboptimally via approximation algorithms. We propose two such algorithms, a "top-down" method and a "bottom-up" method, both of which have complexity that is polynomial in the number of places and transitions. The example of automated guided vehicles shows that the two algorithms, especially the "bottom-up" method, work almost as well as the MIP-based method but with significantly reduced running time. Unlike the OPSS problem, the OTSS problem was shown to be solvable with complexity that is polynomial in the number of places and transitions.

The heuristics used in our approximation algorithms can also be used in other settings. For example, they can be easily adapted to the design of a minimal diagnoser [34] by slightly modifying our approximation algorithms to choose a set of places to put sensors on so that fault transitions can be distinguished immediately. Another application of the heuristics is in rough set theory [35] to compute a reduct with a minimum cardinality of attributes among all possible reducts; in this problem, a reduct is similar to a place sensor configuration in the OPSS problem and attributes are similar to sensors on observable places.

There are several future research directions. First, we plan to consider the general sensor selection problem mentioned in Section III. The goal is to choose a set of place sensors and transition sensors of minimum cardinality such that the system is structurally observable. The general problem is at least as difficult as the OPSS problem. The problem can be solved optimally by exhaustively searching all possibilities. More specifically, one first chooses a place sensor configuration V among 2^{n_1} possible choices, solves the OTSS problem with the fixed place sensor configuration V, obtains the minimum number of transition labels required, and finally adds the number of place sensors and the number of transition labels; the solution selected for the general sensor selection problem is the one with the minimum sum among all 2^{n_1} choices. Though it is not clear how to transform the problem into an integer programming problem due to the heterogeneity of place sensor configurations and transition labeling functions, one might be able to modify the heuristics in Section VII to approximate the optimal solution to this general sensor selection problem.

We also plan to investigate a relaxed version of the OPSS problem (based on the work in [32]) in which we allow multiple states and select a minimum number of place sensors to guarantee that the number of possible states can increase no faster than a given function that is polynomial in the length of the observation sequence.

Another research direction is to consider a variation of the

TABLE III
SIMULATION RESULTS FOR OPSS WITH COSTS

	Top-down Method			Bottom-up Method			MIP-based Method		
i	time (s)	# sensors	cost	time (s)	# sensors	cost	time (s)	# sensors	cost
1	0.391	18	7	0.938	38	3	0.125	23	3
2	0.375	18	24	2.297	36	37	1.781	19	20
3	0.390	18	45	2.109	21	42	5.250	17	36
4	0.375	18	103	2.015	19	98	13.266	17	90
5	0.391	18	193	2.000	18	186	74.266	17	178

OTSS problem, in which only physically close transitions can share the same label. More specifically, the following constraints can be imposed on transition sensors: i) there are only d types of transition sensors $\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_d$; and ii) each type $\mathcal{T}_i$ covers a subset of observable transitions while some transitions may not be covered and some transitions may be covered by more than one type of sensors. We believe the approaches to solve the OPSS problem proposed in this paper could be useful to solve this constrained OTSS problem, and we plan to investigate them in the near future.

Finally, we plan to analyze performance guarantees for the "bottom-up" method. Note that we can reduce the OPSS problem to the set cover problem and solve the constructed set cover problem using a standard greedy algorithm (refer to [36] for the algorithm and its performance guarantee) which selects the subset that can cover the most uncovered elements so far in the universe. This greedy algorithm guarantees a place sensor configuration with the number of place sensors within $OPT * H_q$, where OPT is the minimum number of place sensors used in an optimal solution, q is the parameter in Problem 5 and $H_q = 1 + \frac{1}{2} + ... + \frac{1}{q} = \mathcal{O}(\ln q)$. Directly establishing a performance guarantee for the "bottom-up" method (perhaps in comparison to the solution provided by the set cover greedy algorithm) is an interesting research question.

APPENDIX A
EQUIVALENCE BETWEEN STRUCTURAL OBSERVABILITY AND K-DELAYED STRUCTURAL OBSERVABILITY

We now establish the equivalence between structural observability and K-delayed structural observability.

Proposition 5 Given a place sensor configuration V and a labeling function L, a partially observed Petri net Q being structurally observable is equivalent to Q being K-delayed structurally observable.

Proof: Structural observability $\Longrightarrow$ K-delayed structural observability. If structural observability holds, K-delayed structural observability also holds by definition.

K-delayed structural observability $\Longrightarrow$ structural observability. Suppose $K \geq 1$ (the case for $K = 0$ is trivial). If K-delayed structural observability holds, then the system state at time step i may not be determined uniquely based on the observation sequence generated by transitions up to time step i but can be determined after no more than K additional transition firings. In other words, some of the possible states at time step i given the observations up to time step i vanish due to lack of tokens in certain places (i.e., there are not enough tokens to enable firing sequences up to time step $i + K$). However, these states will still be possible even after K time steps if enough tokens are added in the initial marking while generating the same observation. Since the property has to hold for an arbitrary initial state, we have reached a contradiction. Therefore, the system state at time step i has to be uniquely determined by observations up to time step i, which implies that structural observability holds. ■

APPENDIX B
PROOF OF PROPOSITION 3

To determine transition distinguishability using Proposition 2, for each label $e \in \Sigma \cup \{\varepsilon\}$ satisfying $|T_e| \geq 2$, we need to check whether their corresponding columns in D_V^e are pairwise different. In the worst case, we need to invoke $||V|| \times \binom{|T_e|}{2}$ comparisons. For ε, we need to also check whether any of its transitions corresponds to a zero vector, which needs $||V|| \times |T_\varepsilon|$ additional comparisons. Let $\sum_{e \in \Sigma \cup \{\varepsilon\}, |T_e| \geq 2} |T_e| = m_1 \leq m$. Then

$$\begin{aligned} \#\text{comparisons} &= ||V|| \times |T_\varepsilon| + ||V|| \times \sum_{e \in \Sigma \cup \{\varepsilon\}, |T_e| \geq 2} \binom{|T_e|}{2} \\ &= ||V|| \times \left(|T_\varepsilon| - \frac{m_1}{2} + \sum_{e \in \Sigma \cup \{\varepsilon\}, |T_e| \geq 2} \frac{|T_e|^2}{2} \right) \\ &\leq n \times \left(m - \frac{m_1}{2} + \frac{{m_1}^2}{2} \right) \leq \frac{nm^2 + nm}{2} \end{aligned} \tag{2}$$

where Eq. (2) follows from the fact that $\sum_{i=1}^n a_i^2 \leq (\sum_{i=1}^n a_i)^2$ for positive a_i's, $||V|| \leq n$ and $|T_\varepsilon| \leq m$. Therefore, transition distinguishability can be determined with complexity $\mathcal{O}(nm^2)$.

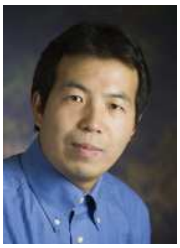
ACKNOWLEDGMENT

The authors would like to thank anonymous reviewers for very insightful and astute comments regarding this paper and earlier conference versions (namely [37], [38]) of parts of this paper.

REFERENCES

[1] P. J. Ramadge and W. M. Wonham, "The control of discrete event systems," *Proceedings of the IEEE*, vol. 77, pp. 81–98, Jan. 1989.
[2] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*. Boston, USA: Kluwer, 1999.
[3] R. Debouk, S. Lafortune, and D. Teneketzis, "On an optimization problem in sensor selection," *Discrete Event Dynamic Systems: Theory and Applications*, vol. 12, pp. 417–445, Oct. 2002.
[4] T. Yoo and S. Lafortune, "NP-completeness of sensor selection problems arising in partially observed discrete-event systems," *IEEE Transactions on Automatic Control*, vol. 47, pp. 1495–1499, Sep. 2002.

[5] K. R. Rohloff, S. Khuller, and G. Kortsarz, "Approximating the minimal sensor selection for supervisory control," *Discrete Event Dynamic Systems: Theory and Applications*, vol. 16, pp. 143–170, Jan. 2006.
[6] S. Jiang, R. Kumar, and H. E. Garcia, "Optimal sensor selection for discrete-event systems with partial observation," *IEEE Transactions on Automatic Control*, vol. 48, pp. 369–381, Mar. 2003.
[7] A. Giua, "Petri net techniques for supervisory control of discrete event systems," in *Proc. First International Workshop on Manufacturing and Petri Nets*, Osaka, Japan, Jun. 1996, pp. 1–30.
[8] J. Esparza and M. Nielsen, "Decidability issues for Petri nets — a survey," *Bulletin of the European Association for Theoretical Computer Science*, vol. 52, pp. 245–262, 1994.
[9] L. Aguirre-Salas, "Sensor selection for observability in interpreted Petri nets: a genetic approach," in *Proc. of 42nd IEEE Conf. on Decision and Control*, Hawaii, USA, Dec. 2003, pp. 3760–3765.
[10] L. E. Holloway and B. H. Krogh, "Synthesis of feedback control logic for a class of controlled Petri nets," *IEEE Transactions on Automatic Control*, vol. 35, pp. 514–523, May 1990.
[11] R. K. Boel, L. Ben-Naoum, and V. V. Breusegem, "On forbidden state problems for a class of controlled Petri nets," *IEEE Transactions on Automatic Control*, vol. 40, pp. 1717–1731, Oct. 1995.
[12] L. E. Holloway, X. Guan, and L. Zhang, "A generalization of state avoidance policies for controlled Petri nets," *IEEE Transactions on Automatic Control*, vol. 41, pp. 804–816, Jun. 1996.
[13] R. S. Sreenivas, "On the existence of supervisory policies that enforce liveness in discrete-event dynamic systems modeled by controlled Petri nets," *IEEE Transactions on Automatic Control*, vol. 42, pp. 928–945, Jul. 1997.
[14] H. Chen, "Net structure and control logic synthesis of controlled Petri nets," *IEEE Transactions on Automatic Control*, vol. 43, pp. 1446–1450, Oct. 1998.
[15] G. Stremersch and R. K. Boel, "Decomposition of the supervisory control problem for Petri nets under preservation of maximal permissiveness," *IEEE Transactions on Automatic Control*, vol. 46, pp. 1490–1496, Sep. 2001.
[16] A. Ghaffari, N. Rezg, and X. Xie, "Feedback control logic for forbidden-state problems of marked graphs: Application to a real manufacturing system," *IEEE Transactions on Automatic Control*, vol. 48, pp. 18–29, Jan. 2003.
[17] F. Basile, C. Carbone, and P. Chiacchio, "Feedback control logic for backward conflict free choice nets," *IEEE Transactions on Automatic Control*, vol. 52, pp. 387–400, Mar. 2007.
[18] L. Zhang and L. E. Holloway, "Forbidden state avoidance in controlled Petri nets under partial observation," in *Proc. of 33rd Annual Allerton Conference on Communications, Control, and Computing*, Monticello, USA, Oct. 1995, pp. 146–155.
[19] A. Giua and C. Seatzu, "Observability of place/transition nets," *IEEE Transactions on Automatic Control*, vol. 47, pp. 1424–1437, Sep. 2002.
[20] Z. Achour, N. Rezg, and X. Xie, "Supervisory control of partially observable marked graphs," *IEEE Transactions on Automatic Control*, vol. 49, pp. 2007–2011, Nov. 2004.
[21] Y. Ru, M. P. Cabasino, A. Giua, and C. N. Hadjicostis, "Supervisor synthesis for discrete event systems with arbitrary forbidden state specifications," in *Proc. of 47th IEEE Conf. on Decision and Control*, Cancun, Mexico, Dec. 2008, pp. 1048–1053.
[22] A. Giua, C. Seatzu, and F. Basile, "Observer-based state-feedback control of timed Petri nets with deadlock recovery," *IEEE Transactions on Automatic Control*, vol. 49, pp. 17–29, Jan. 2004.
[23] M. Oishi, H. Inseok, and C. Tomlin, "Immediate observability of discrete event systems with application to user-interface design," in *Proc. of the 42nd IEEE Conf. on Decision and Control*, Maui, USA, Dec. 2003, pp. 2665–2672.
[24] T. Murata, "Petri nets: Properties, analysis and applications," *Proceedings of the IEEE*, vol. 77, pp. 541–580, Apr. 1989.
[25] Y. Ru and C. N. Hadjicostis, "Fault diagnosis in discrete event systems modeled by partially observed Petri nets," *Discrete Event Dynamic Systems: Theory and Applications*, vol. 19, pp. 551–575, Dec. 2009.
[26] I. Rivera-Rangel, A. Ramírez-Treviño, L. I. Aguirre-Salas, and J. Ruiz-León, "Geometrical characterization of observability in interpreted Petri nets," *Kybernetika*, vol. 41, no. 5, pp. 553–574, 2005.
[27] C. M. Özveren and A. S. Willsky, "Invertibility of discrete-event dynamic systems," *Mathematics of Control, Signals, and Systems*, vol. 5, pp. 365–390, Dec. 1992.
[28] M. R. Garey and D. S. Johnson, *Computers and Intractability — A Guide to the Theory of NP-Completeness*. San Francisco, USA: Freeman, 1979.
[29] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi, *Complexity and Approximation: Combinatorial Optimization Problems and their Approximability Properties*. New York, USA: Springer-Verlag, 1999.
[30] M. Berkelaar, K. Eikland, and P. Notebaert. (2006, Sep.) Open source (mixed-integer) linear programming system (version 5.5.0.8). [Online]. Available: http://sourceforge.net/projects/lpsolve
[31] M. Brodie, I. Rish, and S. Ma, "Optimizing probe selection for fault localization," in *Proc. of 12th Int. Workshop on Distributed Systems: Operations & Management*, Nancy, France, Oct. 2001, pp. 1147–1157.
[32] Y. Ru and C. N. Hadjicostis, "Bounds on the number of markings consistent with label observations in Petri nets," *IEEE Transactions on Automation Science and Engineering*, vol. 6, pp. 334–344, Apr. 2009.
[33] J. O. Moody and P. J. Antsaklis, *Supervisory Control of Discrete Event Systems Using Petri Nets*. Kluwer Academic Publishers, 1998.
[34] D. Lefebvre and C. Delherm, "Diagnosis of DES with Petri net models," *IEEE Transactions on Automation Science and Engineering*, vol. 4, pp. 114–118, Jan. 2007.
[35] Z. Pawlak, *Rough Sets — Theoretical Aspects of Reasoning about Data*. Kluwer Academic Publishers, 1991.
[36] V. V. Vazirani, *Approximation Algorithms*. Springer, 2004.
[37] Y. Ru and C. N. Hadjicostis, "Optimal sensor selection for structural observability in discrete event systems modeled by Petri nets," in *Proc. of First IFAC Workshop on Dependable Control of Discrete Systems*, Paris, France, Jun. 2007.
[38] ——, "Approximating optimal sensor selection for structural observability in discrete event systems modeled by Petri nets," in *Proc. of 46th IEEE Conf. on Decision and Control*, New Orleans, USA, Dec. 2007, pp. 1892–1897.



Yu Ru (S'06) received the B.E. degree in industrial automation in 2002, and the M.S. degree in control theory and engineering in 2005, both from Zhejiang University, Hangzhou, China. He is currently working towards the Ph.D. degree in the Department of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign, Urbana.

His research focuses on discrete event systems and Petri net theory.

Christoforos N. Hadjicostis (M'99, SM'05) received S.B. degrees in Electrical Engineering, in Computer Science and Engineering, and in Mathematics, the M.Eng. degree in Electrical Engineering and Computer Science in 1995, and the Ph.D. degree in Electrical Engineering and Computer Science in 1999, all from the Massachusetts Institute of Technology, Cambridge, MA.

In 1999 he joined the Faculty at the University of Illinois at Urbana-Champaign where he served as assistant and then associate professor with the Department of Electrical and Computer Engineering, the Coordinated Science Laboratory, and the Information Trust Institute. Since 2007, Dr. Hadjicostis has been with the Department of Electrical and Computer Engineering at the University of Cyprus, where he is currently serving as Department Chair. His research focuses on fault diagnosis and tolerance in distributed dynamic systems; error control coding; monitoring, diagnosis and control of large-scale discrete event systems; and applications to network security, anomaly detection, energy distribution systems, medical diagnosis, biosequencing, and genetic regulatory models.