satisfying both MAC1 and MAC2, one may apply the algorithm described in Case 1 to K_1'. □

Note that when G is a scalar automaton and K is a scalar specification, MAC2 for K is trivially true, so this theorem establishes the generalization of the standard result for the scalar case to the MA product case.

ACKNOWLEDGMENT

The authors would like to acknowledge useful conversations with Prof. M. Guay concerning Example 1 in Section II.

REFERENCES

[1] ARENA Software Package [Online]. Available: http://www.rockwellsoftware.com
[2] A. Arnold, *Finite Transition Systems*. Upper Saddle River, NJ: Prentice-Hall, 1994.
[3] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*. Norwell, MA: Kluwer, 1999.
[4] R. Fagin, J. Halperin, Y. Moses, and Y. Moshe, *Reasoning about Knowledge*. Cambridge, MA: MIT Press, 1995.
[5] P. Hubbard and P. E. Caines, "Initial investigations of hierarchical supervisory control for multi-agent systems," in *Proc. 38th IEEE Conf. Decision Control*, vol. 3, Phoenix, AZ, 1999, pp. 2218–2223.
[6] P. Hubbard, "Hierarchical supervisory control systems," Ph.D. dissertation, Dept. Elect. Comp. Eng., McGill Univ., Montreal, QC, Canada, 1999.
[7] M. Heymann, "Concurrency and discrete event control," *IEEE Control Syst. Mag.*, vol. 10, no. 4, pp. 103–112, Apr. 1990.
[8] C. Hoare, *Communicating Sequential Processes*. Upper Saddle River, NJ: Prentice-Hall, 1985, International Series in Computer Science.
[9] J. Hartmanis and R. E. Stearns, *Algebraic Structure Theory of Sequential Machines*. Upper Saddle River, NJ: Prentice-Hall, 1966.
[10] R. Kumar and V. K. Garg, *Modeling and Control of Logical Discrete Event Systems*. Norwell, MA: Kluwer, 1995.
[11] T. Kam, T. Villa, R. Brayton, and A. Sangiovanni-Vincentelli, *Synthesis of Finite State Machines: Functional Optimization*. Norwell, MA: Kluwer, 1997.
[12] Y. Li and W. M. Wonham, "Control of vector discrete-event systems I base model," *IEEE Trans. Autom. Control*, vol. 38, no. 8, pp. 1214–1227, Aug. 1993.
[13] ——, "Concurrent vector discrete event systems," *IEEE Trans. Autom. Control*, vol. 40, no. 4, pp. 628–637, Apr. 1995.
[14] A. Pattavina, "Nonblocking architectures for ATM switching," *IEEE Commun. Mag.*, vol. 31, no. 2, pp. 38–48, Feb. 1993.
[15] P. J. Ramadge and W. M. Wonham, "Supervisory control of a class of discrete event processes," *SIAM J. Control Optim.*, vol. 25, pp. 206–230, 1987.
[16] ——, "The control of discrete event systems," *Proc. IEEE*, vol. 77, no. 1, pp. 81–98, Jan. 1989.
[17] I. Romanovski, "Multi-agent product systems: Analysis, synthesis and control," Ph.D. dissertation, Dept. Elect. Comp. Eng., McGill Univ., Montreal, QC, Canada, 2003.
[18] I. Romanovski and P. E. Caines, "On vector trajectory specifications for multi-agent product systems," in *Proc. 40th IEEE Conf. Decision Control*, Orlando, FL., 2001, pp. 2333–2334.
[19] ——, "Multi-agent products and trajectory specifications in supervisory control theory," in *Proc. Amer. Control Conf.*, Anchorage, AK, May 2002, pp. 722–723.
[20] A. W. Roscoe, *The Theory and Practice of Concurrency*. Upper Saddle River, NJ: Prentice-Hall, 1998.
[21] K. Rudie and J. Willems, "The computational complexity of decentralized discrete-event control problems," *IEEE Trans. Autom. Control*, vol. 40, no. Jul., pp. 1313–1319, 1995.
[22] W. M. Wonham. Notes on control of discrete-event systems. [Online]. Available: http://www.control.utoronto.ca/cgi-bin/dldes.cgi
[23] W. M. Wonham and P. J. Ramadge, "On the supremal controllable sublanguage of a given language," *SIAM J. Control Optim.*, vol. 25, pp. 637–659, 1987.
[24] G. Yin and Q. Zhang, Eds., *Mathematics of Stochastic Manufacturing Systems*. Providence, RI: AMS, 1997.

On Minimal Representations of Petri Net Languages

Ramavarapu S. Sreenivas

***Abstract*—Given a measure of size of a labeled Petri net, we consider the existence of a procedure that takes as input a description of an arbitrary, labeled Petri net, and returns a description of a (possibly different) labeled Petri net with the smallest size that generates the same language as the input. We refer to such procedures as *minimization procedures*. In this note, we investigate the existence of minimization procedures for a variety of measures. We show that these procedures cannot exist for Petri net languages for a large class of measures. However, for families of Petri net languages where *controllability* (cf. [15]), and consequently language-containment, is decidable [16], there can be minimization procedures for a restricted class of measures. After showing that minimization procedures for a family of measures are intractable for languages generated by bounded Petri nets, it is argued that a similar conclusion has to be reached for any family of Petri net languages that includes the family of regular languages for which there are minimization procedures.**

***Index Terms*—Discrete-event systems, Petri nets, supervisory control.**

I. INTRODUCTION

Given an arbitrary, deterministic, finite-state automaton M that generates a language $L(M) \subseteq \Sigma^*$, there exists a procedure for finding the automaton $\widehat{M}$, with the smallest number of states such that $L(\widehat{M}) = L(M)$. The theoretical underpinnings for this procedure is provided by the *Myhill–Nerode Theorem*, and the procedure can be found in most texts on formal languages (cf. [5, Sec. 3.4]). In this note, we consider the analogous problem for labeled Petri nets. This motivated by the fact that the computational effort involved in testing the existence (and synthesis) of supervisory policies for an instance of a supervisory control problem can be significantly reduced when the plant- and desired-behaviors are presented in their minimal forms.

Before we proceed, we have to define how we intend to measure the size of a labeled Petri net. In the context of finite-state automata, it is naturally captured by the number of states in the automata. In the context of Petri nets the choice is not obvious. Intuitively, we expect the notion of size to include the number of places, transitions, arcs and tokens of the initial marking of a Petri net. For instance, we could use the maximum (or the product), of the number of places, number of transitions, number of arcs and the sum of initial token distribution, as an indicator of the size. There might be instances when the number of unbounded places is important, and should be included in the definition of size. Each notion of size can be viewed as a measure over the set of descriptions of labeled Petri nets. Given a description of a labeled Petri net, the measure returns an integer which is a quantitative descriptor of size. If Δ represents the family of descriptions of labeled Petri nets, we can view the measure $\mathcal{M}$ as a total function, $\mathcal{M} : \Delta \to \mathcal{N}$, where $\mathcal{N}$ is the set of nonnegative integers.

We view a *minimization procedure* as a function $\Gamma : \Delta \to \Delta$, where for $M_1 \in \Delta$, if $\Gamma(M_1) = M_2$, then (i) $L(M_1) = L(M_2)$, and (ii)

Manuscript received May 28, 2003; revised February 22, 2005, August 8, 2005, and October 26, 2005. Recommended by Associate Editor A. Giua. This work was supported in part by the National Science Foundation under Grants ECS-0000938 and ECS-04268321, and in part by the Office of Naval Research under Grant N00014-99-1-0696. Portions of this work has appeared in the Proceedings of the 6th International Workshop on Discrete Event Systems, October 2002, Zaragoza, Spain.

The author is with the Coordinated Science Laboratory and the Department of General Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61801 USA (e-mail: rsree@uiuc.edu).

Digital Object Identifier 10.1109/TAC.2006.875026

$\forall M_3 \in \Delta$ such that $L(M_3) = L(M_1), \mathcal{M}(M_2) \leq \mathcal{M}(M_3)$; where $L(M) \subseteq \Sigma^*$ is the language generated by $M \in \Delta$.

In this note, we show that there cannot be a minimization procedure for the family of PN languages for a large collection of measures. However, if we restricted attention to descriptions of labeled PNs where *controllability* (cf. [15]), and consequently language-containment is decidable [16], and if the measure satisfies some additional requirements, then it is possible to find minimization procedures. We then restrict our attention to a family of measures where the minimal labeled PN that produces the empty language is the PN with no places and no transitions. For this family of measures we show that the minimization procedure for languages generated by bounded PNs is intractable. So, a similar conclusion has to be reached for any family of PN languages that contains the class of regular languages.

The rest of the note is organized as follows. Section II contains the notations and definitions. Section III presents the main result of this note. Section IV contains the motivation and implications of these results, followed by the conclusions, which are presented in Section V.

II. Notations and Definitions

A *Petri net* (PN) $N = (\Pi, T, \Phi, \mathbf{m}^0)$ is an ordered 4-tuple, where $\Pi = \{p_1, \ldots, p_n\}$ is a set of n *places*, $T = \{t_1, \ldots, t_m\}$ is a collection of m *transitions*, $\Phi \subseteq (\Pi \times T) \cup (T \times \Pi)$ is a set of *arcs*, $\mathbf{m}^0 : \Pi \to \mathcal{N}$ is the *initial marking function* (or the *initial marking*), and $\mathcal{N}$ is the set of nonnegative integers. The state of a PN is given by the *marking* $\mathbf{m} : \Pi \to \mathcal{N}$ which indicates the distribution of *tokens* in each place. For a given marking $\mathbf{m}$, a transition $t \in T$ is said to be *enabled* if $\forall p \in^\bullet t$, $\mathbf{m}(p) \geq 1$, where $^\bullet x := \{y \mid (y, x) \in \Phi\}$. For a given marking $\mathbf{m}$ the set of enabled transitions is denoted by the symbol $T_e(N, \mathbf{m})$. An enabled transition $t \in T_e(N, \mathbf{m})$ can *fire*, which changes the marking $\mathbf{m}$ to $\widehat{\mathbf{m}}$ according to the equation $\widehat{\mathbf{m}}(p) = \mathbf{m}(p) - \mathrm{card}(p^\bullet \cap \{t\}) + \mathrm{card}(^\bullet p \cap \{t\})$, where $x^\bullet := \{y \mid (x, y) \in \Phi\}$ and the symbol $\mathrm{card}(\bullet)$ is used to denote the cardinality of the set argument. In this note we do not consider simultaneous firing of multiple transitions.

A string of transitions $\omega = t_1 t_2 \cdots t_k$, where $t_i \in T$ $(i \in \{1, 2, \ldots, k\})$ is said to be a *valid firing sequence* starting from the marking $\mathbf{m}$, if: 1) the transition t_1 is enabled under the marking $\mathbf{m}$, and 2) for $i \in \{1, 2, \ldots, k-1\}$ the firing of the transition t_i produces a marking under which the transition t_{i+1} is enabled. The marking resulting from the firing of $\omega \in T^*$ starting from the initial marking $\mathbf{m}^0$ is denoted by $\delta(\mathbf{m}^0, \omega) \in \mathcal{N}^n$.

Given an initial marking $\mathbf{m}^0$ the set of *reachable markings* for $\mathbf{m}^0$ denoted by $\Re(N, \mathbf{m}^0)$, is defined as the set of markings generated by all valid firing sequences starting with marking $\mathbf{m}^0$ in the PN N. A place $p \in \Pi$, is said to be *unbounded* if for every $k \in \mathcal{N}$, there is a marking $\mathbf{m} \in \Re(N, \mathbf{m}^0)$ such that $\mathbf{m}(p) > k$. For an arbitrary $\mathbf{m} \in \mathcal{N}^n$, it is of interest to know if $\mathbf{m} \in \Re(N, \mathbf{m}^0)$. This is known as the *reachability problem* (cf. [13, Ch. 5]), which is decidable [11], [8], but has a nonprimitive recursive complexity.

In some instances, such as in Section IV of this note, the set of arcs of a PN $N = (\Pi, T, \Phi, \mathbf{m}^0)$ might have weights, $\mathcal{W} : \Phi \to \mathcal{N}$, associated with them. Such PN's are referred to as *general* PNs. A transition $t \in T$ in a general PN is enabled under a marking $\mathbf{m}$ if $\forall p \in^\bullet t, \mathbf{m}(p) \geq \mathcal{W}((p, t))$. The firing of an enabled transition $t \in T$ under the marking $\mathbf{m}$ results in a marking $\widehat{\mathbf{m}}$, where $\widehat{\mathbf{m}}(p) = \mathbf{m}(p) - \mathcal{W}(p, t) + \mathcal{W}(t, p)$. PNs defined at the beginning of this section are general PNs where the weight associated with each arc is unit. Such PNs are referred to as *ordinary PNs*. It is known that any general PN can be converted into an equivalent ordinary PN (cf. [13, Sec. 5.3]). That is, every general (ordinary) PN can be transformed into a ordinary (general) PN, in such a way that a reachability problem in the original general (ordinary) PN can be reduced to a reachability problem in its equivalent ordinary (general) PNs.

Following Peterson (cf. [13, Sec. 6.3.3]), a language $L \subseteq \Sigma^*$ is said to be an *L-type Petri net language* (*P-type Petri net language*) if there is a Petri net $N = (\Pi, T, \Phi, \mathbf{m}^0)$, a *labeling function* $\alpha : T \to \Sigma$, and a finite set of *final-markings* $\mathcal{F} \subseteq \mathcal{N}^n$ such that $L = \{\alpha(\omega) \in \Sigma^* \mid \delta(\mathbf{m}^0, \omega) \in \mathcal{F}\}$ $(L = \{\alpha(\omega) \in \Sigma^* \mid \delta(\mathbf{m}^0, \omega)$ is defined$\})$. A language $L \subseteq \Sigma^*$ is said to be a *deterministic*, *P*-type language, if there is a Petri net $N = (\Pi, T, \Phi, \mathbf{m}^0)$, a *labeling function* $\alpha : T \to \Sigma$, such that $L = \{\alpha(\omega) \in \Sigma^* \mid \delta(\mathbf{m}^0, \omega)$ is defined$\}$ and $\forall \mathbf{m} \in \Re(N, \mathbf{m}^0)$, $\forall t_1, t_2 \in T_e(N, \mathbf{m}), (t_1 \neq t_2) \Rightarrow (\alpha(t_1) \neq \alpha(t_2))$ [12]. If necessary, this notion of determinism can be extended by requiring that the firing of either of two similarly labeled transitions at a marking should result in the same marking. That is, $\forall \mathbf{m} \in \Re(N, \mathbf{m}^0), \forall t_1, t_2 \in T_e(N, \mathbf{m}), (\alpha(t_1) = \alpha(t_2)) \Rightarrow (\delta(\mathbf{m}, t_1) = \delta(\mathbf{m}, t_2))$.

We use the term *labeled Petri net* to denote the triple $M = (N, \alpha, \mathcal{F})$, where $N = (\Pi, T, \Phi, \mathbf{m}^0)$ is the underlying PN, $\alpha : T \to \Sigma$ is the labeling function, and $\mathcal{F}$ is the finite-set of final-markings. For *P*-type languages the set $\mathcal{F}$ is irrelevant, so it can be set to the empty set, or dropped entirely from the definition.

The relationships between *L*- and *P*-type languages can be found in Peterson's text (cf. [13, Sec. 6.3.4]). For each of these classes, the *language-containment problem* is undecidable [4]. That is, given two languages $L_1, L_2 \in \Sigma^*$ from any one of these classes, it is impossible to test if $L_1 \subseteq L_2$. Also, for any $L \subseteq \Sigma^*$ from any one of these classes, it is not possible to decide if L is a regular language [19], [6]. [2] identifies a hierarchy of free-labeled and deterministic PN languages that is richer than the *P*- and *L*-type languages identified previously. In this reference, the set of final-markings, $\mathcal{F} \subseteq \mathcal{N}^n$, is not necessarily finite. In specific, for the family of PN languages, where the labeling of PN is deterministic, and the set $\mathcal{F} \subseteq \mathcal{N}^n$ is *semi-linear* (cf. [9, p. 127]), it is shown that the containment problem is decidable. This result is established through a reduction to the PN reachability problem. As a consequence, any decision procedure for containment for this family of PN languages as at least as hard as the PN reachability problem [11], [8].

Algorithmic complexity (AC) (or, *Kolmogorov complexity*) of a string $\omega \in \Sigma^*$ is the length of the shortest description of a Turing machine that produces the string ω as an output. Theorem 2.1 states that the algorithmic complexity of an arbitrary string is not computable. An intuitive sketch of the proof of this result can be found in [9]; detailed proofs of the same can be found in [10].

Theorem 2.1: Kolmogorov–Chaitin–Solomonoff Incomputability Theorem [9]: There is no algorithm for computing the AC of an arbitrary string (to even within a fixed-constant).

We use the set $\Delta \subseteq \widehat{\Sigma}^*$ to denote descriptions of labeled Petri nets. The format of these descriptions, and the definition of the set $\widehat{\Sigma}$ is not important. The important thing to keep in mind is that there is an automated procedure that can generate labeled PNs out of these descriptions. It would help to view these descriptions in $\Delta \subseteq \widehat{\Sigma}^*$ as strings when input to a (universal) Turing Machine produces an appropriate description of a labeled PN. As long as we are interested in algorithmic complexity, the enunciation of this automated procedure is also unimportant. As pointed out in [10], [9], it is possible to append a preamble of fixed-length to the descriptions of the labeled PNs such that an universal computational machine can generate the labeled PNs using the chosen automated procedure. This adds a fixed, constant to the algorithmic complexity. For large descriptions, this fixed amount becomes insignificant, and is ignored in the literature on AC [10].

To make the *use/mention distinction*, when we use the notation $M \in \Delta$ in this note we are referring to the labeled PN represented by the string $\langle M \rangle \in \widehat{\Sigma}^*$. We use the symbol $L_l(M)$ and $L_p(M)$ to denote the *L*-, and *P*-type languages generated by M. A *measure* $\mathcal{M}$ is a recursive

function $\mathcal{M} : \Delta \rightarrow \mathcal{N}$ that returns an integer that is indicative of size for any member of Δ.

A *minimization procedure for L-type languages* is a recursive function $\Gamma_l : \Delta \rightarrow \Delta$, where for $\langle M_1 \rangle \in \Delta$, if $\Gamma_l(\langle M_1 \rangle) = \langle M_2 \rangle$, then i) $L_l(M_1) = L_l(M_2)$, and ii) $\forall \langle M_3 \rangle \in \Delta$, such that $L_l(M_3) = L_l(M_1)$, $\mathcal{M}(\langle M_2 \rangle) \leq \mathcal{M}(\langle M_3 \rangle)$. Minimization procedures for *P*-type languages, namely $\Gamma_p : \Delta \rightarrow \Delta$ are similarly defined. Although we do not explicitly consider them in this note, minimization procedures can be envisioned for the hierarchy of deterministic PN languages identified in [2]. In the next section, we consider different candidates for the measure $\mathcal{M} : \Delta \rightarrow \mathcal{N}$ and explore the existence of minimization procedures with specific properties.

III. Main Results

We consider the existence of a minimization procedure for *L*-type languages $\Gamma_l : \Delta \rightarrow \Delta$, for different choices of the measure $\mathcal{M} : \Delta \rightarrow \mathcal{N}$. Our first choice is $\mathcal{M}_1(\langle M \rangle) = \mid \langle M \rangle \mid$, where $\mid \bullet \mid$ denotes the length of the string argument. In other words, for this choice we are interested in a minimization procedure that when presented with a description of a labeled PN M_1 as an input, presents as output the description of (a possibly different) labeled PN M_2, such that i) $L_l(M_1) = L_l(M_2)$, and ii) $\forall \langle M_3 \rangle \in \Delta$, such that $L_l(M_3) = L_l(M_1)$, $\mid \langle M_2 \rangle \mid \leq \mid \langle M_3 \rangle \mid$. That is, the minimization procedure $\Gamma_l(\bullet)$, when supplied with a description of M_1, returns the description of (a possibly different) labeled PN M_2 such that $\langle M_2 \rangle$ is the shortest among all descriptions of labeled PNs that produce the same language as M_1. The following theorem states that $\Gamma_l(\bullet)$ as described previously cannot exist.

Theorem 3.1: There is no minimization procedure $\Gamma_l : \Delta \rightarrow \Delta$ for *L*-type languages for the measure $\mathcal{M}_1 : \Delta \rightarrow \mathcal{N}$ such that $\mathcal{M}_1(\langle M \rangle) = \mid \langle M \rangle \mid$ for $\langle M \rangle \in \Delta$.

Proof (Contradiction): Let us suppose there is a minimization procedure for the proposed measure. Let us suppose $\langle M_2 \rangle \in \Delta$ be a string such that for some $\langle M_1 \rangle \in \Delta$, $\Gamma_l(\langle M_1 \rangle) = \langle M_2 \rangle$. It follows from the definition of minimization procedures that, for the proposed measure, $AC(\langle M_2 \rangle) = \mid \langle M_2 \rangle \mid$. To see this, note that if there is a string ω, where $\mid \omega \mid < \mid \langle M_2 \rangle \mid$, when input to a (universal) Turing Machine produces $\langle M_2 \rangle$ as output, then it stands to reason that the proposed minimization procedure would have produced ω in lieu of $\langle M_2 \rangle$.

Note that the set of (minimal representations of) *L*-type languages, $\mathcal{S} \subseteq \Delta (\subseteq \widehat{\Sigma}^*)$, is countably-large. So, there is a recursive function $f : \Sigma^* \rightarrow \mathcal{S}$ that represents a bijective-map between Σ^* and $\mathcal{S}$. Let us suppose the program $\phi \in \Sigma^*$ effectively computes $f : \Sigma^* \rightarrow \mathcal{S}$. So, we can use the string $\langle f(\gamma), \phi \rangle$ as a code, or as a short-hand-notation, for an arbitrary string $\gamma \in \Sigma^*$. So, $AC(\gamma) \leq \mid f(\gamma) \mid + \mid \phi \mid$. For lengthy strings, where the additive constant $\mid \phi \mid$ can be ignored, we have $AC(\gamma) \approx \mid f(\gamma) \mid$, as we know that there can be no shorter description of the string $f(\gamma)$. This violates Theorem 2.1. ■

The above result also holds for minimization procedures for the *P*-type language of a PN that uses the measure $\mathcal{M}_1$ defined above. Essentially, the above proof is a restatement of the *Kolmogorov–Chaitin–Solmonoff Incomputability Theorem* [9] which states that the shortest-description of any object (not just labeled PNs) is incomputable.

Earlier, the only requirement we placed on the set of descriptions $\Delta \subseteq \widehat{\Sigma}^*$ is that they should be strings when input to a Universal Turing machine produced an appropriate description of a labeled PN as the output. This would imply that the set of descriptions are recursive-enumerable, and in turn there can be several undecidable results that one could generate as a consequence of Rice's Theorem (cf. [5, Sec. 8.4]). In the remainder, we consider the existence of minimization procedures when there are additional restrictions on the set of descriptions $\Delta \subseteq \widehat{\Sigma}^*$. In specific, we assume the descriptions $\Delta \subseteq \widehat{\Sigma}^*$ are strongly typed so that the following claim can be made.

1) The set $\Delta \subseteq \widehat{\Sigma}^*$ is recursive, that is, given an arbitrary string in $\widehat{\Sigma}^*$, there is a computational procedure $\Lambda : \widehat{\Sigma}^* \rightarrow \{1, 0\}$, that determines if the string is a valid description of a labeled PN. If $\Lambda(\omega) = 1$ ($\Lambda(\omega) = 0$), we suppose the string ω is a valid (an invalid) description of a labeled PN.
2) Given any two strings in Δ it is possible to decide if these strings represent the same labeled PN.

For a string $\langle \omega \rangle \in \widehat{\Sigma}^*$, and an ordering of $\widehat{\Sigma}^*$, the position of $\langle \omega \rangle$ in the ordering is denoted by the symbol $\Psi(\langle \omega \rangle)$. We assume the function $\Psi : \widehat{\Sigma}^* \rightarrow \mathcal{N}$ is recursive. We say the measure $\mathcal{M}$ is *primitive recursive* with respect to an ordering if this ordering is recursive, and $\exists f : \mathcal{N} \rightarrow \mathcal{N}$, which is recursive, such that

$$\forall n \in \mathcal{N}, \max\{\Psi(\langle \omega \rangle) \mid \langle \omega \rangle \in \mathcal{M}^{-1}(n)\} < f(n) < \infty.$$

Essentially, the previous statement says that the set $\{\langle M \rangle \in \Delta \mid \mathcal{M}(\langle M \rangle) = n\}$ can be constructed by looking at the first $f(n)$ strings in the ordering of $\widehat{\Sigma}^*$.

Let us now consider a measure $\mathcal{M}_2 : \Delta \rightarrow \mathcal{N}$ that takes into account the number of unbounded places in the underlying PN of a labeled PN. In particular, let us suppose that if $\mathcal{M}_2(\langle M \rangle) = 0$ for some $\langle M \rangle \in \Delta$ then it implies that the number of unbounded places in the underlying PN of M is zero. For instance, this property is true if $\mathcal{M}_2(\langle M \rangle)$ can be written as

$$\mathcal{M}_2(\langle M \rangle) = (\#\text{unbounded-places in the underlying PN of } M) \times \mathcal{M}_i(\langle M \rangle)$$

for some measure $\mathcal{M}_i : \Delta \rightarrow \mathcal{N}$. The following theorem, which is a direct consequence of the undecidability of regularity for *L*- and *P*-type languages [19], [6] and the fact that boundedness of a PN is decidable [14], shows that minimization procedures for this class of measures cannot exist.

Theorem 3.2: There can be no minimization procedures for the *L*- and *P*-type languages for the measure $\mathcal{M}_2 : \Delta \rightarrow \mathcal{N}$ with the property:

$$(\mathcal{M}_2(\langle M \rangle) = 0) \Leftrightarrow (\#\text{unbounded places in the underlying PN of } M = 0).$$

Proof: For the class of measures in the statement of the theorem, the *L*-type (*P*-type) language generated by a labeled PN M is regular if and only if $\mathcal{M}_2(\Gamma_l(\langle M \rangle)) = 0$ ($\mathcal{M}_2(\Gamma_p(\langle M \rangle)) = 0$). If there is a minimization procedure for the aforementioned measure, then we have an effective test for regularity of an arbitrary *L*- or *P*-type language, as the boundedness of the underlying PN of $\Gamma_l(\langle M \rangle)$ ($\Gamma_p(\langle M \rangle)$)is decidable [14]. Since it is not possible to decide the regularity of an arbitrary *L*- or *P*-type language [19], [6] the result follows. ■

In the following, we consider a family of measures, $\mathcal{M}_3 : \Delta \rightarrow \mathcal{N}$, where no two descriptions of labeled PNs that map onto the same integer produce the same *L*-type (*P*-type) language. The following result shows that minimization procedures cannot exist for this class of measures.

Theorem 3.3: There can be no minimization procedure $\Gamma_l : \Delta \rightarrow \Delta$ ($\Gamma_p : \Delta \rightarrow \Delta$) for a measure $\mathcal{M}_3 : \Delta \rightarrow \mathcal{N}$ that satisfies the requirement $\forall n \in \mathcal{N}, \forall \langle M_1 \rangle, \langle M_2 \rangle \in \mathcal{M}_3^{-1}(n), (L_l(M_1) =$

$L_l(M_2)) \Rightarrow (\langle M_1\rangle = \langle M_2\rangle)$ and $\forall n \in \mathcal{N}, \forall \langle M_1\rangle, \langle M_2\rangle \in \mathcal{M}_3^{-1}(n), (L_p(M_1) = L_p(M_2)) \Rightarrow (\langle M_1\rangle = \langle M_2\rangle)$.

Since no two descriptions of labeled PNs that map onto the same integer produce the same language, it follows that $(L_l(M_1) = L_l(M_2)) \Leftrightarrow (\langle\Gamma_l(\langle M_1\rangle)\rangle = \langle\Gamma_l(\langle M_2\rangle)\rangle)$. The theorem follows as a consequence of the undecidability of the language equivalence problem for *L*- and *P*-type PN languages (cf. [4, Th. 8.2]), and the fact that the equivalence of any two, finite-length strings in $\widehat{\Sigma}^*$ is decidable.

Let us now turn our attention to features that we might want, but cannot have, in the minimization procedures $\Gamma_l : \Delta \to \Delta$ ($\Gamma_p : \Delta \to \Delta$) for arbitrary measures. The following theorem, like Theorem 3.3, is also a direct consequence of the undecidability of the language equivalence problem for *L*- and *P*-type PN languages (cf. [4, Th. 8.2]), and the fact that the equivalence of any two, finite-length strings in Δ is decidable.

Theorem 3.4: For an arbitrary measure $\mathcal{M} : \Delta \to \mathcal{N}$, there can be no minimization procedure $\Gamma_l : \Delta \to \Delta$ ($\Gamma_p : \Delta \to \Delta$) that satisfies the requirement: $\forall M_1, M_2 \in \Delta$, $(L_l(M_1) = L_l(M_2)) \Rightarrow (\Gamma_l(\langle M_1\rangle) = \Gamma_l(\langle M_2\rangle))$ and $\forall M_1, M_2 \in \Delta$, $(L_p(M_1) = L_p(M_2)) \Rightarrow (\Gamma_p(\langle M_1\rangle) = \Gamma_p(\langle M_2\rangle))$.

Consider a partition of

$$\Delta = \Delta_0 \cup \Delta_1 \cup \cdots \Delta_i \cup \cdots (\forall i, j \in \{0, 1, 2, \ldots\}, \Delta_i \cap \Delta_j = \emptyset)$$

where Δ_i is a collection of descriptions of labeled PNs $M = (N, \alpha, \mathcal{F})$ where $N = (\Pi, T, \Phi, \mathbf{m}^0)$, $\alpha : T \to \Sigma$, $\mathrm{card}(\mathcal{F}) < \infty$, and the description of any labeled PN $(N, \alpha, \mathcal{F})$ that satisfies the requirement – (#places in N) $\leq i$, (#transitions in N) $\leq i$, (#arcs in N) $\leq i$, $(\sum_{j=1}^{n} \mathbf{m}_j^0) \leq = i$, $\mathrm{card}(\mathcal{F}) \leq i$, and $\forall \mathbf{m}_j^f \in \mathcal{F}$, $(\sum_{j=1}^{n} \mathbf{m}_j^f) \leq = i$, can be found in one of the sets Δ_j, $(j \in \{0, 1, \ldots, i\})$.

Since

$$\mathrm{card}(\Delta_i) \ll 2^i \times 2^i \times 2^{2i^2} \times \Omega \times \mathrm{card}(\Sigma)^i \times \Omega^i < \infty$$
$$\text{where} \quad \Omega = 1 + \sum_{k=1}^{i} \binom{i+k-1}{k-1}$$

the enumeration of elements of Δ where the elements of Δ_i are lexicographically enumerated before any element of Δ_{i+1}, presents a well-defined order. It is not difficult to see that the measure for *P*-type (i.e., $\mathcal{F} = \emptyset$) languages, as shown in the first equation at the bottom of the page. is primitive recursive with respect to the order defined above. The following observation is about minimization procedures for measures that are primitive recursive with respect to a specific ordering.

Let $\mathcal{M}_4 : \Delta \to \mathcal{N}$ be a measure that is primitive recursive with respect to an ordering of Δ. Also, let $\widetilde{\Delta} \subseteq \Delta$ be a recursive subset of representations of labeled PNs where language-equality is decidable. For instance, $\widetilde{\Delta}$ could be the set of descriptions of deterministic, labeled PNs. From the results in [12], and the function $\Lambda(\bullet)$ in Section II, we know the set $\widetilde{\Delta}$ is recursive. Note that the well-defined ordering on Δ induces a well-defined ordering on $\widetilde{\Delta}$.

The following result shows that there is a minimization procedure for the *L*- and *P*-type languages of the descriptions in $\widetilde{\Delta}$. In the following, $\langle\Gamma(\bullet)\rangle$ can also be viewed as a procedure that returns a description of a finite-set (as opposed to a singleton-member) of minimal, labeled PNs for an appropriate argument.

Theorem 3.5: For a recursive subset of descriptions $\widetilde{\Delta} \subseteq \Delta$, where language-equality is decidable (that is, $\forall \langle M_1\rangle, \langle M_2\rangle \in \widetilde{\Delta}$, there is a procedure that decides if $L(M_1) = L(M_2)$), and a measure $\mathcal{M}_4 : \Delta \to \mathcal{N}$ that is primitive recursive with respect to an ordering of $\widetilde{\Delta}$, it is possible to find a minimization procedure $\Gamma_l : \widetilde{\Delta} \to \widetilde{\Delta}$ ($\Gamma_p : \widetilde{\Delta} \to \widetilde{\Delta}$) for *L*-type (*P*-type) languages of labeled PN descriptions in $\widetilde{\Delta}$

Proof: Suppose we are given a description of M, then $\forall n \leq \mathcal{M}_4(M)$, the elements of the set $\mathcal{M}_4^{-1}(n) \cap \widehat{\Delta}$ can be effectively and finitely enumerated as the measure $\mathcal{M}_4$ is primitive recursive with respect to the given ordering. For each $\widehat{M} \in \mathcal{M}_4^{-1}(n)$, we can test if $L_l(M) = L_l(\widehat{M})$ ($L_p(M) = L_p(\widehat{M})$). The descriptions $\widehat{M}$ that satisfy this test are placed in a set Φ. The set Φ will be finite in size, and any fixed member of the set $\{\widehat{M} \in \Phi \mid \forall \widehat{M} \in \Phi, \mathcal{M}_4(\widehat{M}) \leq \mathcal{M}_4(\widetilde{M})\}$ can be presented as the minimal equivalent of M with respect to the measure $\mathcal{M}_4$. ∎

Note that the minimization procedure outlined in the proof has the property

$$\forall M_1, M_2 \in \widetilde{\Delta}, (L_l(M_1) = L_l(M_2)) \Rightarrow (\langle\Gamma_l(M_1)\rangle = \langle\Gamma_l(M_2)\rangle).$$

Suppose, we concerned ourselves with measures $\mathcal{M}_5 : \Delta \to \mathcal{N}$ such that the set $\mathcal{M}_5^{-1}(n)$ is finite, for any $n \in \mathcal{N}$. For instance, the measures shown in the second or third equations at the bottom of the page, satisfy this property. Let us also suppose we are only interested in minimization procedures for this class measures that have the following "uniqueness" property: $\forall \langle M_1\rangle, \langle M_2\rangle \in \widetilde{\Delta}$

$$(L_l(M_1) = L_l(M_2)) \Rightarrow (\langle\Gamma_l(M_1)\rangle = \langle\Gamma_l(M_2)\rangle).$$

The decidability of language equivalence for the family of descriptions $\widetilde{\Delta} \subseteq \Delta$ is both necessary and sufficient for the existence of a minimization procedure. This is formally stated in Theorem 3.6.

$$\mathcal{M}(\langle N\rangle) = \max\left\{(\#\text{places in } N), (\#\text{transitions in } N), (\#\text{arcs in } N), \left(\sum_{j=1}^{n} \mathbf{m}_j^0\right)\right\}$$

$$\max\{(\#\text{places in } M),\ (\#\text{ transitions in } M),\ (\#\text{arcs in } M),\ (\#\text{tokens in the initial marking})\}$$

$$\max\{(\#\text{places in } M),\ (\#\text{transitions in } M),\ (\#\text{tokens in the initial marking})\}$$

Theorem 3.6: Suppose $\widetilde{\Delta} \subseteq \Delta$ is a recursive subset of descriptions of a family of labeled PNs, and $\mathcal{M}_5 : \Delta \to \mathcal{N}$ is a recursively computable measure that is primitive recursive with respect to an given ordering of $\widetilde{\Delta}$ then there is a minimization procedure $\Gamma_l : \widetilde{\Delta} \to \widetilde{\Delta}$ ($\Gamma_p : \widetilde{\Delta} \to \widetilde{\Delta}$) for *L*-type (*P*-type) languages of labeled PN descriptions in $\widetilde{\Delta}$ that satisfies the property

$$\forall M_1, M_2 \in \widetilde{\Delta}, (L_l(M_1) = L_l(M_2)) \Rightarrow (\langle \Gamma_l(M_1) \rangle = \langle \Gamma_l(M_2) \rangle)$$

if and only if language-equality is decidable for the family of descriptions in $\widetilde{\Delta} \subseteq \Delta$.

Proof: If language-containment is decidable for the family of descriptions $\widetilde{\Delta}$, then from Theorem 3.5, we know that there is a minimization procedure $\Gamma_l(\bullet)$ with the property

$$\forall M_1, M_2 \in \widetilde{\Delta}, (L_l(M_1) = L_l(M_2)) \Rightarrow (\langle \Gamma_l(M_1) \rangle = \langle \Gamma_l(M_2) \rangle).$$

The converse implication is obvious. ∎

The property of *controllability* [15] should be decidable in modeling paradigms intended for discrete-event systems. Since the test for language-containment reduces to a test for controllability [16], from Theorem 3.6 we infer that minimization procedures for the class of measures defined in the statement of Theorem 3.6 exist for modeling paradigms intended for discrete-event systems.

We now turn our attention to specific measures. Consider a measure $\mathcal{M}_6 : \Delta \to \mathcal{N}$, where

$$\mathcal{M}_6(M) = (\#\text{places in } M) + (\#\text{transitions in } M) + (\#\text{arcs in } M) + (\#\text{tokens in the initial marking}).$$

Let us suppose we are interested in minimization procedures, $\Gamma_l : \Delta \to \Delta$, for *L*-type languages for the previous measure. It can be shown that if $L_l(M) = \Sigma^*$ for some $M \in \Delta$, then $\Gamma_l(\langle M \rangle) = \langle \widehat{M} \rangle$, where $\widehat{M} = (N, \alpha, \emptyset)$, $N = (\emptyset, \{t_1, t_2, \ldots, t_{\mathrm{card}(\Sigma)}\}, \emptyset, \epsilon)$, and $\alpha(\bullet)$ is any bijection from T to Σ, and ϵ is the null-function. Note that there are $\mathrm{card}(\Sigma)!$ many choices for the bijection from T to Σ, but for each choice we obtain the same labeled PN. We can test if $L_l(\widetilde{M}) = \Sigma^*$ for an arbitrary labeled PN $\widetilde{M}$ by checking if $\Gamma_l(\langle \widetilde{M} \rangle)$ is the same as the description of one of the $\mathrm{card}(\Sigma)!$ many choices that describe $\widehat{M}$.

In [6], it is shown that it is possible to test if a *P*-type language is equivalent to a given prefix-closed regular language. That is, given a labeled PN, and a deterministic, finite-state automaton where are states are final states (cf. [9, p. 51]), it is possible to test if the *P*-type language of the labeled PN is the same as the language generated by the automaton. Therefore, it is possible to test if $L_p(\langle M \rangle) = \Sigma^*$ for an arbitrary labeled PN M. However, testing if $L_l(\langle M \rangle) = \Sigma^*$ is undecidable.[1] To see this we first note that the problem of deciding "$L = \Sigma^*$?" is undecidable when L is a language accepted by a nondeterministic, reversal-bounded, one-counter machine [1]. Next, we observe that *one-way blind counter machines* [3] are equivalent to nondeterministic, reversal-bounded, one-counter machines. The undecidability of "$L_l(\langle M \rangle) = \Sigma^*$?" follows that from the fact that the more general class of *Partially blind counter machines* [3] are equivalent to PNs. Consequently, it follows that the minimization procedure referred to previously cannot exist for the measure $\mathcal{M}_6$.

IV. Discussion

We turn our attention to the practical motivation for seeking the minimal representation of PN languages. In the context of supervisory control, testing the controllability [15] of a language with respect to another plays a very important role in the synthesis of solutions to various supervisory control problems. Typically, the languages are represented within a chosen paradigm. If these representations are minimal, then the computational effort expended in testing controllability is also minimal. Alternately, it is possible to expend significant effort in testing the controllability of nonminimal representations. [18] contains examples of the savings in computation time that can be realized when minimal representations are used in supervisory control.

Reference [18] also contains a few illustrative examples where the minimality of specific instances of labeled PNs are certified using the results of Theorem 3.6. For instance, the *deterministically labeled, sequential Petri net* (DSPN) [17] shown in Fig. 1 is minimal with respect to the measure shown in the equation at the bottom of the page. That is, there is no DSPN with a smaller measure that produces the same language as the one shown in Fig. 1.

In all the examples in [18], the process of certifying minimality involved the time-consuming process of exhaustive enumeration of all labeled PNs with smaller or equal measure to the labeled PN at hand, till an equivalent labeled PN was found. This leads to us ask if there could be an efficient minimization procedure for the families of the PNs considered in [18] and other families of PN languages. We now show that there can be no efficient minimization procedure even for languages generated by bounded PNs.[2] The following observation plays a crucial role in arriving at this conclusion.

Observation 4.1: The reachability problem for bounded PNs reduces to the nonemptiness of the *L*-type language generated by bounded, labeled PNs.

Proof: Given an arbitrary bounded PN $N = (\Pi, T, \Phi, \mathbf{m}^0)$, and an n-dimensional ($n = \mathrm{card}(\Pi)$), nonnegative, integer-valued vector $\mathbf{m} \in \mathcal{N}^{\mathrm{card}(\Pi)}$, let us suppose we are interested in checking if $\mathbf{m} \in \Re(N, \mathbf{m}^0)$.

We can convert N into a (bounded) labeled PN $(N, \alpha, \mathcal{F})$, where $\mathcal{F} = \{\mathbf{m}\}$ and $\alpha : T \to T$ is the identity function (i.e., $\alpha(t) = t$). It is not hard to see that the *L*-type language generated by the $(N, \alpha, \mathcal{F})$ is nonempty if and only if $\mathbf{m} \in \Re(N, \mathbf{m}^0)$. ∎

The PN with no places and no transitions is the minimal labeled PN that generates the empty-language under the the measure $\mathcal{M}_6(\bullet)$ of the previous section. So, if we were to test the emptiness of the *L*-type language generated by a bounded, labeled PN. We could input its description to a minimization procedure for the measure $\mathcal{M}_6(\bullet)$ for *L*-type languages generated by bounded PN's, and we could check if the minimal description was the PN with no places and transitions. This, observation 4.1, and the fact that deciding reachability for bounded PNs is PSPACE-hard (cf. [7, Cor. 3.3]), suggests that the computational effort involved in the minimization for languages generated by bounded, labeled PNs is PSPACE-hard. This observation holds for any measure under which the PN with no places and no transitions is the minimal labeled PN that generates the empty-set.

[1] We thank an anonymous referee of the note for this result.

[2] This family of languages is identical to the family of regular languages.

$$\max\{(\#\text{places}), (\#\text{transitions}), (\#\text{arcs}), (\#\text{tokens in the initial marking})\}.$$

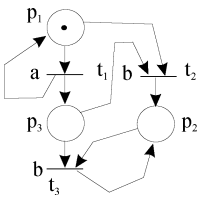

Fig. 1. Minimal, deterministically labeled, sequential PN under the measure

$$\max\{(\#\text{places}),\ (\#\text{ transitions}),\ (\#\text{arcs})\ (\#\text{tokens in the initial marking})\}.$$

We turn our attention to a version of the above observation for *P*-type languages. For this, we consider the measure $\mathcal{M}_7 : \Delta \to \mathcal{N}$, where

$$\mathcal{M}_7(M) = (\#\text{places in } M) \times (\#\text{transitions in } M) \times (\#\text{arcs in } M).$$

Under this measure, all transitions in a minimal representation of a PN language must fire at least once. There are other measures where this property is also true, but we will stick to $\mathcal{M}_7(\bullet)$ for purposes of illustration.

Given a bounded, PN $N = (\Pi, T, \Phi, \mathbf{m}^0)$, and an n-dimensional ($n = \text{card}(\Pi)$), nonnegative, integer-valued vector $\mathbf{m} \in \mathcal{N}^{\text{card}(\Pi)}$, let us suppose we are interested in deciding the coverability of $\mathbf{m}$. That is, we are interested in the existence of $\widehat{\mathbf{m}} \in \Re(N, \mathbf{m}^0)$, such that $\widehat{\mathbf{m}} \geq \mathbf{m}$, componentwise.

We can use the minimization procedure for languages generated by bounded, labeled PNs under the measure $\mathcal{M}_7(\bullet)$, to solve the aforementioned coverability problem. To see this, add a new and unused transition $\widehat{t}$ to the PN N, such that $\widehat{t}^{\bullet} = \emptyset$. There is an arc, of weight $\mathbf{m}_i$ from each place p_i to $\widehat{t}$. This *general* PN (cf. [13, page 125]) is then converted into an equivalent *ordinary* PN using the procedure in [13, Sec. 5.3]. The details of this routine, yet laborious, transformation is skipped for brevity. It is important to note that in the resulting ordinary PN, the transition $\widehat{t}$ can fire if and only if there is a reachable marking of N that covers $\mathbf{m}$. As before, the ordinary PN can be viewed as a free-labeled, bounded, marked PN that represents a *P*-type language. This description of the bounded, labeled PN can be input to the minimization procedure for the measure $\mathcal{M}_7(\bullet)$, and we can test the existence of a cover for $\mathbf{m}$ by checking if there is a transition labeled $\widehat{t}$ in the minimal representation. Since the coverability problem for bounded PNs is PSPACE-hard (cf. [7, Cor. 3.2]), it follows that the computational effort involved in producing the minimal representation of a language generated by a bounded, labeled PN is PSPACE-hard. So, it is highly unlikely that there are efficient minimization procedures for any family of PN languages, where language-containment is decidable, and this family includes the class of regular languages.

V. Conclusion

Motivated by the minimization procedures for deterministic, finite-state automata (cf. [5, Sec. 3.4], in this note we looked at similar procedures for labeled PNs. For deterministic, finite-state automata the notion of size is effectively captured by the number of states in the automaton. For labeled Petri nets there is no obvious choice of measure that denotes size. As a consequence, in this note we looked at families of measures with different properties that capture different notions of size. We found that for a large collection of these families there cannot be a minimization procedure. However, if we restricted attention to descriptions of labeled PNs where language-containment is decidable, and measures that have a finite-inverse, it is possible to find minimization procedures.

The property of *controllability* [15] should be decidable for any modeling paradigm intended for discrete-event systems. Since the test for language-containment reduces to a test for controllability [16], minimization procedures exist for measures with a finite-inverse in families of labeled PNs used to model discrete-event systems. We showed that there cannot be efficient minimization procedures for a class of measures for languages generated by bounded, labeled PNs. This result suggests that efficient minimization procedures cannot exist for any class of PN languages that contain the class of regular languages.

Acknowledgment

The authors would like to thank the anonymous referees for their suggestions on improving the readability of the previous versions of this note.

References

[1] B. S. Baker and R. V. Book, "Reversal-bounded multipushdown machines," *J. Comput. Syst. Sci.*, vol. 8, pp. 315–332, 1974.
[2] S. Gaubert and A. Giua, "Petri net languages and infinite subsets of $\mathcal{N}^m$," *J. Comput. Syst. Sci.*, vol. 59, pp. 373–391, 1999.
[3] S. A. Greibach, "Remarks on blind and partially blind one-way multicounter machines," *Theoret. Comput. Sci.*, vol. 7, pp. 311–324, 1978.
[4] M. H. T. Hack, "Petri net languages," Mass. Inst. Technol., Cambridge, MA, Tech. Rep. 159, 1976.
[5] J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation*. Reading, MA: Addison-Wesley, 1979, Series in Computer Science.
[6] P. Jančar, J. Esparza, and F. Moller, "Petri nets and regular processes," *J. Comput. Syst. Sci.*, vol. 59, pp. 476–503, 1999.
[7] N. D. Jones, L. H. Landweber, and Y. E. Lien, "Complexity of some problems in Petri nets," *Theoret. Comput. Sci.*, vol. 4, pp. 277–299, 1977.
[8] S. R. Kosaraju, *Decidability Reachability Vector Addition Syst.*, pp. 267–281, 1982.
[9] J. P. Lewis, "Large limits to software estimation," *ACM Software Eng. Notes*, vol. 26, no. 4, pp. 54–59, Jul. 2001.
[10] M. Li and P. Vitànyi, *An Introduction to Kolmogorov Complexity and Its Applications*, 2nd ed. New York: Springer-Verlag, 1997, Graduate Texts in Computer Science.
[11] E. W. Mayr, "An algorithm for the general Petri net reachability problem," *SIAM J. Comput.*, vol. 13, pp. 441–460, 1984.
[12] E. Pelz, "Closure properties of deterministic Petri nets," in *STACS 87*. Berlin, Germany: Springer-Verlag, 1987, vol. 247, Lecture Notes in Computer Science, pp. 371–382.
[13] J. L. Peterson, *Petri Net Theory and the Modeling of Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1981.
[14] C. Rackoff, "The covering and boundedness problems for vector addition systems," *Theoret. Comput. Sci.*, vol. 6, no. 2, pp. 223–231, Apr. 1978.
[15] P. J. G. Ramadge and W. M. Wonham, "The control of discrete event systems," *Proc. IEEE*, vol. 77, no. 1, pp. 81–98, Jan. 1989.
[16] R. S. Sreenivas, "On a weaker notion of controllability of a language K with respect to a language L," *IEEE Trans. Autom. Control*, vol. 38, no. 9, pp. 1446–1447, Sep. 1993.
[17] ——, "On asympotically efficient solutions for a class of supervisory control problems," *IEEE Trans. Autom. Control*, vol. 41, no. 12, pp. 1736–1750, Dec. 1996.
[18] ——, "On minimal representations of Petri net languages: Theory and illustrative examples," Univ. Illinois at Urbana-Champaign, Coordinated Science Laboratory, Urbana, IL, Res. Rep. UILU-ENG 05-2202, 2005.
[19] R. Valk and G. Vidal-Naquet, "Petri nets and regular languages," *J. Comput. Syst. Sci.*, vol. 3, pp. 299–325, 1981.