satisfying both MAC1 and MAC2, one may apply the algorithm described in Case 1 to $K'$. \\

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


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 $\hat{M}$, with the smallest number of states such that $L(\hat{M}) = L(M)$. 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 $\Delta$ 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 $M_i \in \Delta$, if $\Gamma(M_i) = M_k$, then (i) $L(M_i) = L(M_k)$, and (ii)
∀M₂ ∈ Δ such that L(M₂) = L(M₁), M₂(M₂) ≤ M(M₁); where 
L(M)  ∩ Σ* is the language generated by M ∈ Δ.

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 con-
trollability (cf. [15]), and consequently language-containment is decid-
able [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 transi-
tions. For this family of measures we show that the minimization pro-
cedure 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 re-
results, followed by the conclusions, which are presented in Section V.

II. NOTATIONS AND DEFINITIONS

A Petri net (PN) N = (Π, T, Φ, m₀) is an ordered 4-tuple, where 
Π = {p₁, . . . , pₙ} is a set of n places, T = {t₁, . . . , tₘ} is a col-
lection of m transitions, Φ ⊆ (Π × T) ∪ (T × Π) is a set of arcs, 
m₀ : Π → N is the initial marking function (or the initial marking),
and N is the set of nonnegative integers. The state of a PN is given
by the marking m : Π → N which indicates the distribution of
tokens in each place. For a given marking m, a transition t ∈ T is said to be enabled if ∀p ∈ e⁺ t, m(p) ≥ 1, where e⁺ t := {y | (y, x) ∈ Φ}. For a given marking m the set of enabled transitions is
denoted by the symbol Tₑ(N, m). An enabled transition t ∈ Tₑ(N, m)
can fire, which changes the marking m to m̂ according to the equa-
tion m̂(p) = m(p) − card(e⁺ t ∩ {p}) + card(e⁻ t ∩ {t}), where 
e⁻ t := {y | (x, y) ∈ Φ} and the symbol card() is used to denote the cardinality of the set argument. In this note we do not consider si-
multaneous firing of multiple transitions.

A string of transitions ω = t₁t₂ . . . tₖ, where tᵢ ∈ T (i ∈ {1, 2, . . . , k}) is said to be a valid firing sequence starting from
the marking m₀, if: 1) the transition t₁ is enabled under the marking m₀, and 2) for i ∈ {1, 2, . . . , k − 1} the firing of the transition tᵢ produces a marking under which the transition tᵢ₊₁ is enabled. The marking resulting from the firing of ω ∈ T* starting from the initial marking m₀ is denoted by m(ω, m₀) ∈ N*.

Given an initial marking m₀ the set of reachable markings for m₀ denoted by R(N, m₀), is defined as the set of markings generated by all valid firing sequences starting with marking m₀ in the PN N. A place p ∈ Π is said to be unbounded if for every k ∈ N, there is a marking m ∈ R(N, m₀) such that m(p) > k. For an arbitrary m ∈ N*, it is of interest to know if m ∈ R(N, m₀). 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 = (Π, T, Φ, m₀) might have weights, W : Φ → N, associated with them. Such PNs’ are referred to as general PNs. A transition t ∈ T in a general PN is enabled under a marking m if ∀p ∈ e⁺ t, m(p) ≥ W(p, t). The firing of an enabled transition t ∈ T under the marking m results in a marking m̂, where m̂(p) = m(p) − W(p, t) + W(t, p). PN 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 ⊆ Σ* is said to be an L-type Petri net language (P-type Petri net language) if there is a Petri net N = (Π, T, Φ, m₀), a labeling function α : T → Σ, and a finite set of final-markings F ⊆ Σn such that L = {a(ω) ∈ Σ* | δ(m₀, ω) ∈ F} (L = {a(ω) ∈ Σ* | δ(m₀, ω) is defined}). A language L ⊆ Σ* is said to be a deterministic, P-type language, if there is a Petri net N = (Π, T, Φ, m₀), a labeling function α : T → Σ, such that L = {a(ω) ∈ Σ* | δ(m₀, ω) is defined} and ∀m ∈ R(N, m₀), ∀t₁, t₂ ∈ T(N, m), (t₁ ≠ t₂) ⇒ (α(t₁) ≠ α(t₂))[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, ∀m ∈ R(N, m₀), ∀t₁, t₂ ∈ T(N, m), α(t₁) = α(t₂) ⇒ δ(m, t₁) = δ(m, t₂).

We use the term labeled Petri net to denote the triple M = (N, α, F), where N = (Π, T, Φ, m₀) is the underlying PN, α : T → Σ is the labeling function, and F is the finite-set of final-markings. For P-type languages the set 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 L₁, L₂ ⊆ Σ* from any one of these classes, it is impossible
to decide if L₁ ⊆ L₂. Also, for any L₁ ⊆ Σ* from any one of these classes, it is not possible to decide if L₁ 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, F ⊆ Σn, is not necessarily finite. In specific, for the family of PN languages, where the labeling of PN is deterministic, and the set F ⊆ Σn 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 ω ∈ Σ* 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 Δ ⊆ Σ* to denote descriptions of labeled Petri nets.
The format of these descriptions, and the definition of the set Σ* is not important. The important thing to keep in mind is that there is an automatic procedure that can generate labeled PNs out of these descriptions. It would help to view these descriptions in Δ ⊆ Σ* 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 unim-
portant. 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 ∈
Δ in this note we are referring to the labeled PN represented by the
string (M), ∈ Σ*. We use the symbol L(M) and L₀(M) to denote the
L- and P-type languages generated by M. A measure M is a recursive
function $\mathcal{M} : \Delta \to \mathcal{N}$ that returns an integer that is indicative of size for any member of $\Delta$. A minimization procedure for $L$-type languages is a recursive function $\Gamma_1 : \Delta \to \Delta$, where for $(\langle M \rangle_1) \in \Delta$, if $\Gamma_1((\langle M \rangle_1)) = (\langle M \rangle_2)$, then i) $L_1(\langle M \rangle_1) = L_1(\langle M \rangle_2)$, and ii) $\forall \langle M \rangle_3 \in \Delta$, such that $L_1(\langle M \rangle_3) = L_1(\langle M \rangle_2)$, $\mathcal{M}(\langle M \rangle_3) \leq \mathcal{M}(\langle M \rangle_2)$. Minimization procedures for $P$-type languages, namely $\Gamma_P : \Delta \to \Delta$ 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 $\mathcal{M} : \Delta \to \mathcal{N}$ 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 $\Gamma_1 : \Delta \to \Delta$, for different choices of the measure $\mathcal{M} : \Delta \to \mathcal{N}$. Our first choice is $\mathcal{M}_1(\langle M \rangle) \equiv |\langle M \rangle|$, where $|\textit{-}|$ 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) $L_1(\langle M \rangle_1) = L_1(\langle M \rangle_2)$, and ii) $\forall \langle M \rangle_3 \in \Delta$, such that $L_1(\langle M \rangle_3) = L_1(\langle M \rangle_2)$, $|\langle M \rangle_3| \leq |\langle M \rangle_2|$. That is, the minimization procedure $\Gamma_1(\textit{-})$, when supplied with a description of a labeled PN, returns the description of (a possibly different) labeled PN $M_2$ such that the $\langle M \rangle_2$ is the shortest among all descriptions of labeled PNs that produce the same language as $M_1$. The following theorem states that $\Gamma_1(\textit{-})$ as described previously cannot exist.

Theorem 3.1: There is no minimization procedure $\Gamma_1 : \Delta \to \Delta$ for $L$-type languages for the measure $\mathcal{M}_1 : \Delta \to \mathcal{N}$ such that $\mathcal{M}_1(\langle M \rangle_1) \equiv |\langle M \rangle|$ for $(\langle M \rangle_1) \in \Delta$.

Proof (Contradiction): Let us suppose there is a minimization procedure for the proposed measure. Let us suppose $\langle M \rangle_2 \in \Delta$ be a string such that for some $\langle M \rangle_1 \in \Delta$, $\Gamma_1((\langle M \rangle_1)) = (\langle M \rangle_2)$. It follows from the definition of minimization procedures that, for the proposed measure, $A(\mathcal{C}(\langle M \rangle_2)) \equiv |\langle M \rangle_2|$. To see this, note that if there is a string $\omega$, where $\omega \in \Sigma^*$, when input to a (universal) Turing Machine produces $\langle M \rangle_2$ as output, then it stands to reason that the proposed minimization procedure would have produced $\omega$ in lieu of $\langle M \rangle_2$.

Note that the set of (minimal representations) of $L$-type languages, $\Sigma^* \subseteq \Delta \subseteq \Sigma^*$, is countably-large. So, there is a recursive function $f : \Sigma^* \to \Sigma$ that represents a bijective-map between $\Sigma^*$ and $\Delta$. Let us suppose the program $\delta \in \Sigma^*$ effectively computes $f : \Sigma^* \to \Delta$. So, we can use the string $\{\Gamma(\gamma), \delta\}$ as a code, or as a short-hand-notation, for an arbitrary string $\gamma \in \Sigma^*$. So, $A(\mathcal{C}(\gamma)) \leq |\{\Gamma(\gamma), \delta\}| + |\delta|$. For lengthy strings, where the additive constant $|\delta|$ can be ignored, we have $A(\mathcal{C}(\gamma)) \equiv |\Gamma(\gamma)|$, as we know that there can be no shorter description of the string $\Gamma(\gamma)$. 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–Solomonoff 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 $\Delta \subseteq \hat{\Sigma}^*$ 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 $\Delta \subseteq \hat{\Sigma}^*$. In specific, we assume the descriptions $\Delta \subseteq \hat{\Sigma}^*$ are strongly typed so that the following claim can be made.

1) The set $\Delta \subseteq \hat{\Sigma}^*$ is recursive, that is, given an arbitrary string in $\hat{\Sigma}^*$, there is a computational procedure $\Lambda : \hat{\Sigma}^* \to \{0, 1\}$, that determines if the string is a valid description of a labeled PN. If $\Lambda(\omega) = 1$ ($\Lambda(\omega) = 0$), we suppose the string $\omega$ is a valid (an invalid) description of a labeled PN.

2) Given any two strings in $\Delta$ it is possible to decide if these strings represent the same labeled PN.

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

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

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

Let us now consider a measure $\mathcal{M}_2 : \Delta \to \Delta$ 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

$$\mathcal{M}_2(\langle M \rangle) \equiv (\# \text{unbounded places})$$

in the underlying PN of $M \times \mathcal{M}(\langle M \rangle)$ for some measure $\mathcal{M}_1 : \Delta \to \mathcal{N}$. 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 $\mathcal{M}_2 : \Delta \to \mathcal{N}$ with the property:

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

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 $\mathcal{M}_2(\Gamma_1((\langle M \rangle))) = 0$ ($\mathcal{M}_2(\Gamma_P((\langle M \rangle))) = 0$). 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 $\Gamma_1((\langle M \rangle))$ 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, $\mathcal{M}_3 : \Delta \to \mathcal{N}$, 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 $\Gamma_1 : \Delta \to \Delta$ ($\Gamma_P : \Delta \to \Delta$) for a measure $\mathcal{M}_3 : \Delta \to \mathcal{N}$ that satisfies the requirement $\forall n \in \mathcal{N}, \forall \langle M \rangle_1, \langle M \rangle_2 \in \mathcal{M}_3^{-1}(n)$, $L_1(\langle M \rangle_1) = L_1(\langle M \rangle_2)$.
For an arbitrary measure $M : \Delta \rightarrow \mathcal{N}$, there can be no minimization procedure $\Gamma : \Delta \rightarrow \Delta$ (where $\Gamma : \Delta \rightarrow \Delta$) that satisfies the requirement: $\forall M_1, M_2 \in \Delta, (L_p(M_1) = L_p(M_2)) \Rightarrow (\Gamma(L_p(M_1)) = \Gamma(L_p(M_2)))$. Consider a partition of

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

where $\Delta_i$ is a collection of descriptions of labeled PNs $M = (N, \alpha, F)$ where $N = (\Pi, T, \Phi, m^0)$, $\alpha : T \rightarrow \Sigma$, $\text{card}(F) < \infty$, and the description of any labeled PN $(N, \alpha, F)$ that satisfies the requirements $\text{card}(\text{places in } N) \leq i$, $\text{card}(\text{transitions in } N) \leq i$, $(\sum_{j=0}^{i} m_{ij}^0) = i$, $\sum_{j=0}^{i} \text{card}(F) \leq i$, and $\forall i \in F, (\sum_{j=0}^{i} m_{ij}^0) \leq i$, can be found in one of the sets $\Delta_j$, $j \in \{0, 1, 2, \ldots\}$.

Since

$$\text{card}(\Delta_j) \leq 2^i \times 2^i \times 2^{2+i} \times \Omega \times \text{card}(\Sigma)^i \times \Omega^i < \infty$$

where

$$\Omega = 1 + \sum_{k=1}^{\infty} \left(\frac{i+k-1}{k-1}\right)$$

the enumeration of elements of $\Delta_j$ where the elements of $\Delta_i$ are lexicographically enumerated before any element of $\Delta_{i+1}$, presents a well-defined order. It is not difficult to see that the measure for $P$-type (i.e., $F = \emptyset$) 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 $\mathcal{M}_i : \Delta \rightarrow \mathcal{N}$ be a measure that is primitive recursive with respect to an ordering of $\Delta$. Also, let $\Delta' \subseteq \Delta$ be a recursive subset of representations of labeled PNs where language-equality is decidable. For instance, $\Delta$ could be the set of descriptions of deterministic, labeled PNs. From the results in [12], and the function $\Lambda(\bullet \bullet)$ in Section II, we know the set $\Delta$ is recursive. Note that the well-defined ordering on $\Delta$ induces a well-defined ordering on $\Delta'$.

The following result shows that there is a minimization procedure for the $L$- and $P$-type languages of the descriptions in $\Delta$. In the following, $\Lambda(\bullet \bullet)$ 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 3.3**: For a recursive subset of descriptions $\Delta' \subseteq \Delta$, where language-equality is decidable (that is, $\forall (M_1, M_2) \in \Delta'$, there is a procedure that decides if $L(M_1) = L(M_2)$, and a measure $\mathcal{M}_i : \Delta \rightarrow \mathcal{N}$ that is primitive recursive with respect to an ordering of $\Delta$, it is possible to find a minimization procedure $\Gamma : \Delta \rightarrow \Delta$ (where $\Gamma : \Delta \rightarrow \Delta$) for $L$-type ($P$-type) languages of labeled PN descriptions in $\Delta'$.

**Proof**: Suppose we are given a description of $M$, then $\forall n \leq \mathcal{M}_i(M_1) \leq \mathcal{M}_i(M_2)$, the elements of the set $\mathcal{M}_i(\n) \cap \Delta'$ can be effectively and finitely enumerated as the measure $\mathcal{M}_i$ is primitive recursive with respect to the given ordering. For each $\mathcal{M} \in \mathcal{M}_i(\n)$, we can test if $L_i(M) = L_i(\mathcal{M})$, where $L_i(M) = L_i(\mathcal{M})$. The descriptions $\mathcal{M}$ that satisfy this test are placed in a set $\Phi$. The set $\Phi$ will be finite in size, and any fixed member of the set $\mathcal{M} \in \Phi$ such that $\mathcal{M}(\n) \leq \mathcal{M}_i(M)$ can be presented as the minimal equivalent of $M$ with respect to the measure $\mathcal{M}_i$.

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

$$\forall M_1, M_2 \in \Delta', (L_i(M_1) = L_i(M_2)) \Rightarrow (\Gamma(L_i(M_1)) = \Gamma(L_i(M_2)))$$

Suppose, we concerned ourselves with measures $\mathcal{M}_i : \Delta \rightarrow \mathcal{N}$ such that the set $\mathcal{M}_i(\n) \cap \Delta'$ 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: $\forall (M_1, M_2) \in \Delta'$

$$(L_i(M_1) = L_i(M_2)) \Rightarrow (\Gamma(L_i(M_1)) = \Gamma(L_i(M_2)))$$

The decidability of language equivalence for the family of descriptions $\Delta' \subseteq \Delta$ is both necessary and sufficient for the existence of a minimization procedure. This is formally stated in Theorem 3.6.
**Theorem 3.6:** Suppose \( \hat{\Delta} \subseteq \Delta \) is a recursive subset of descriptions of a family of labeled PNs, and \( M_6 : \Delta \rightarrow \mathcal{N} \) is a recursively computable measure that is primitive recursive with respect to an given ordering of \( \hat{\Delta} \) then there is a minimization procedure \( \Gamma_1 : \hat{\Delta} \rightarrow \hat{\Delta} \) for L-type (P-type) languages of labeled PN descriptions in \( \hat{\Delta} \) that satisfies the property

\[
\forall M_1, M_2 \in \hat{\Delta}, (L_i(M_1) = L_i(M_2)) \Rightarrow (\Gamma_1(M_1) = \Gamma_1(M_2))
\]

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

**Proof:** If language-containment is decidable for the family of descriptions \( \Delta \), then from Theorem 3.5, we know that there is a minimization procedure \( \Gamma_1(\bullet) \) with the property

\[
\forall M_1, M_2 \in \Delta, (L_i(M_1) = L_i(M_2)) \Rightarrow (\Gamma_1(M_1) = \Gamma_1(M_2)).
\]

The converse implication is obvious. \( \square \)

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 \( M_6 : \Delta \rightarrow \mathcal{N} \), where

\[
M_6(M) = (\# \text{places in } M) + (\# \text{transitions in } M) + (\# \text{arcs in } M) + (\# \text{tokens in the initial marking})
\]

Let us suppose we are interested in minimization procedures, \( \Gamma_1 : \Delta \rightarrow \hat{\Delta} \), for L-type languages with the previous measure. It can be shown that if \( L_i(M) = \Sigma^* \) for some \( M \in \Delta \), then \( \Gamma_1(M) = (\hat{M}) \), where \( \hat{M} = (N, \alpha, \emptyset, \emptyset, \emptyset, \epsilon) \), and \( \alpha(\bullet) \) is any bijection from \( T \) to \( \Sigma \), and \( \epsilon \) is the null-function. Note that there are \( \text{card}(\Sigma)! \) many choices for the bijection from \( T \) to \( \Sigma \), but for each choice we obtain the same labeled PN. We can test if \( L_i(\hat{M}) = \Sigma^* \) for an arbitrary labeled PN \( \hat{M} \) by checking if \( \Gamma_1(\hat{M}) \) is the same as the description of one of the \( \text{card}(\Sigma)! \) many choices that describe \( \hat{M} \).

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 language generated by the automaton. Therefore, it is possible to test if \( L_i(M) = \Sigma^* \) for an arbitrary labeled PN \( M \). However, testing if \( L_i(M) = \Sigma^* \) 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 non-deterministic, reversal-bounded, one-counter machine [1]. Next, we observe that one-way blind counter machines [3] are equivalent to non-deterministic, reversal-bounded, one-counter machines. The undecidability of \( L_i(M) = \Sigma^* \) follows 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 \( M_6 \).

**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 us to 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 \( N = (\Pi, T, \Phi, m^0) \), and an \( n \)-dimensional \( \alpha = \text{card}(\Pi) \), nonnegative, integer-valued vector \( m \in \Lambda^{\alpha}(\Pi^0) \), let us suppose we are interested in checking if \( m \in R(N, m^0) \).

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

The PN with no places and no transitions is the minimal labeled PN that generates the empty-language under the the measure \( M_6(\bullet) \) 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 \( M_6(\bullet) \) for L-type languages generated by bounded PNs, 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.

---

1This family of languages is identical to the family of regular languages.

2This family of languages is identical to the family of regular languages.
max \{ (\# \text{places}), (\# \text{transitions}), (\# \text{arcs})
\}
\left(\# \text{tokens in the initial marking}\right)\}

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

\[\mathcal{M}_\tau(M) = (\# \text{places in } M) \times (\# \text{transitions in } M) \times (\# \text{arcs in } M)\]

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 \(\mathcal{M}_\tau(\bullet)\) 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 \(\mathbf{m} \in \mathbb{N}^{\text{card}(\Pi)}\), let us suppose we are interested in deciding the coverability of \(\mathbf{m}\). That is, we are interested in the existence of \(\hat{\mathbf{m}} \in \mathbb{R}(\mathcal{N}, \mathbf{m}^0)\), such that \(\hat{\mathbf{m}} \geq \mathbf{m}\), componentwise.

We can use the minimization procedure for languages generated by bounded, labeled PNs under the measure \(\mathcal{M}_\tau(\bullet)\), to solve the aforementioned coverability problem. To see this, add a new and unused transition \(\tilde{\tau}\) to the PN \(N\), such that \(\mathbf{P} = \emptyset\). There is an arc, of weight \(\mathbf{m}_i\), from each place \(p_i\) to \(\tilde{\tau}\). 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 \(\tilde{\tau}\) can fire if and only if there is a reachable marking of \(\mathcal{N}\) 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 \(\mathcal{M}_\tau(\bullet)\), and we can test the existence of a cover for \(\mathbf{m}\) by checking if there is a transition labeled \(\tilde{\tau}\) 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-coverability 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-coverability 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