On the Existence of Supervisory Policies that Enforce Liveness in Discrete-Event Dynamic Systems Modeled by Controlled Petri Nets

Ramavarapu S. Sreenivas, *Member, IEEE*

***Abstract*— We consider discrete-state plants represented by** *controlled Petri nets* **(CtlPN's), where a subset of transitions can be prevented from firing by a supervisor. A transition in a CtlPN can fire at a marking if there are sufficient tokens in its input places and it is permitted to fire by the supervisor. A CtlPN is** *live* **if it is possible to fire any transition from every marking that is reachable under supervision. In this paper we derive a necessary and sufficient condition for the existence of a supervisory policy that enforces liveness in CtlPN's. We show this condition cannot be tested for an arbitrary CtlPN. However, for bounded CtlPN's, or CtlPN's where each transition is individually controllable, we show the existence of a supervisory policy which enforces that liveness is decidable. We also show the existence of a supervisory policy that enforces liveness is necessary and sufficient for the existence of a minimally restrictive supervisory policy.**

***Index Terms*—DEDS, liveness, Petri nets, supervisory control.**

I. Introduction

AREAS that involve resource sharing, like flexible-manufacturing systems, distributed computing, and operations management, require an explicit resource allocation policy that provides clear directives on the equitable distribution of scarce resources. These allocation rules, while being satisfactory on a standalone basis, often fail when interaction among the processes is considered. An anthropomorphic analogy could be the well-intentioned directive of waiting for the other person to use the doorway when two persons arrive simultaneously at either ends. While sensible, this directive creates a deadlock when the individuals on either side apply the *same* directive and wait indefinitely for the other to use the doorway. Analogous situations can occur with greater severity in the above-mentioned application areas. In a sense, it is the study of this phenomenon that is the main focus of this paper. In particular, we study processes modeled by *controlled Petri nets* (CtlPN's) [8], [9]. CtlPN's are extensions of standard *Petri nets* (PN's) [13], [14]. In the following two paragraphs we introduce the key ideas using PN's with the appropriate addendum for CtlPN's.

Manuscript received April 12, 1996; revised January 27, 1997. Recommended by Associate Editor, E. K. P. Chong. This work was supported in part by the UIUC Campus Research Board under Grant RES-BRD-IC-SREENIVAS-1-2-68317 and the National Science Foundation under Grant ECS-9409691.

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).

Publisher Item Identifier S 0018-9286(97)05066-6.

A transition in a PN is *live* if for any reachable marking there exists a valid firing sequence that results in a marking that enables the transition under consideration. If all transitions in a PN are live, then the PN is said to be live. Therefore, a live PN does not deadlock, although the definition of liveness is stronger than just the absence of deadlocks (cf. [2, Sec. 2.3]). A *CtlPN* is derived from a PN by using an additional set of *control places*. *Control arcs* are arcs from control places to transitions in a CtlPN. The control places can be thought of as autonomous input places to transitions with a binary token load determined by an external agent, the supervisor. We refer to the "regular" places (i.e., the places that are not control places) as *state places*. The input place set to an arbitrary transition in a CtlPN can be partitioned into state places and control places. Transitions that (do not) have a control place in the input place set are called *controlled* (*uncontrolled*) transitions. Without loss in generality, we can assume each controllable transition has only one control place in its input place set. A supervisory policy can be viewed as an implicitly defined table that lists transitions that are permitted to fire for each reachable marking of the CtlPN, where the uncontrollable transitions are always permitted to fire. A transition in a CtlPN is *state-enabled* at a given marking if every input place to the said transition has a nonzero token-load. Similarly, a transition in a CtlPN is *control-enabled* at a given marking if the supervisory policy permits the firing of the said transition. A transition in a CtlPN has to be state-enabled and control-enabled to fire. The notion of a valid firing sequence for a given marking is defined accordingly. A transition in a CtlPN is *live* if for every marking reachable under supervision, there exists a valid firing sequence that results in a marking under which the said transition is control-enabled and state-enabled (cf. [14, Sec. 4.1.4, level 4 liveness]). A supervisory policy enforces liveness if every transition in the CtlPN is live under supervision. For a given CtlPN, the underlying PN can be obtained by deleting the control places and the control arcs. If the underlying PN of a CtlPN is live, the trivial policy of control-enabling all transitions under any marking enforces liveness. In a sense, in this paper we concern ourselves with CtlPN's where the underlying PN is not live, and we explore conditions under which a supervisory policy which enforces that liveness exists. We derive a necessary and sufficient condition for the existence of a supervisory policy that enforces liveness in CtlPN's. We show that the existence of a supervisory policy that enforces liveness in arbitrary

CtlPN's is undecidable. However, for bounded CtlPN's, or CtlPN's where each transition is individually controllable, we show the existence of a supervisory policy that enforces liveness is decidable.

We use the paradigm of supervisory control of *discrete-event dynamic systems* (DEDS) to enforce liveness in CtlPN's. The literature in supervisory control of DEDS [15], [16] concerns primarily two classes of problems: *forbidden-state problems* and *forbidden-string problems*. The following discussion concerning forbidden-state problems applies to forbidden-string problems as well. For forbidden state problems, we identify a subset of plant states as a desirable set. A supervisory control policy is a static table that provides a list of controllable transitions to be disabled for each state reachable in the closed-loop. The set of states reachable under supervision is said to be control invariant. A supervisory control policy solves the forbidden-state problem if the set of states reachable in the closed-loop is a subset of the desired set. For various reasons it is of interest to identify the control policy that is minimally restrictive. Ramadge and Wonham provide a general technique to identify the minimally restrictive policy for finite-state systems in their paper [15]. We turn our attention to the problem in this paper. We have a CtlPN, the plant, with an underlying PN that is not live. Its transitions can be temporarily disabled by the supervisor. The objective is to implicitly identify a static table that lists the disabled transitions for the reachable markings under supervision to enforce liveness in the closed loop. The solution to this problem involves three steps: 1) testing the existence of a supervisory policy that enforces liveness; 2) the identification of the set of desirable (legal) states; and 3) investigating the control invariance of the desirable states.

A qualitative specification like liveness could also play a critical role in the quantitative analysis of DEDS (cf. [3] and [7] for details). As an example, consider a stochastic PN simulated using the *generalized semi-Markov process* (GSMP) paradigm (cf. [18, ch. 1]). If the PN is live and the probability density function of the firing time for each transition has an unlimited support, then the set of reachable markings of the stochastic PN will be identical to that of the underlying untimed PN. In addition, for any reachable marking, each enabled transition has a nonzero probability of being assigned the shortest firing time compared to its contending transitions. This suggests the probability of any transition firing infinitely often in any sample path is unity. If the underlying PN is not live, then there will be at least one transition that does not fire infinitely often in every sample path. The supervisory policies introduced in this paper can help rectify any potential problems if the requirement that every transition fires infinitely often almost surely is critical to quantitative control.

In the next section we present a brief overview of PN theory along with the definitions of various terms used in subsequent text. Section III contains examples of CtlPN's that can, and cannot, be made live via supervision. These examples set the stage for the necessary and sufficient condition for the existence of a supervisory policy that enforces liveness and related results in Section V. Section IV contains a survey of relevant work in the literature. In Section VI we revisit the examples of Section III and present some additional insights. Finally, in Section VII we conclude with some suggested future research directions. The Appendix contains an algorithm that is critical to results presented in this paper.

II. Symbols and Definitions

A PN $N = (\Pi, T, \Phi, \mathbf{m}^0)$ is an ordered 4-tuple, where $\Pi = \{p_1, p_2, \cdots, p_n\}$ is a set of n *places*, $T = \{t_1, t_2, \cdots, t_m\}$ is a set of m *transitions*, $\Phi \subseteq (\Pi \times T) \cup (T \times \Pi)$ is a set of *arcs*,[1] $\mathbf{m}^0$: $\Pi \rightarrow \mathcal{N}$ is the *initial marking function* (or the *initial marking*), and $\mathcal{N}$ is the set of nonnegative integers. The *state* of a PN is the marking $\mathbf{m}$: $\Pi \rightarrow \mathcal{N}$ that identifies the number of *tokens* in each place. PN's can represent infinite-state systems as the value of the marking can be unbounded. A marking $\mathbf{m} : \Pi \rightarrow \mathcal{N}$ is sometimes represented by an integer-valued vector $\mathbf{m} \in \mathcal{N}^n$, where the ith component $\mathbf{m}_i$ represents the token load $(\mathbf{m}(p_i))$ of the ith place. The context should suggest the appropriate usage. 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 ${}^\bullet x := \{y | (y, x) \in \Phi\}$. For a given marking $\mathbf{m}$, the set of enabled transitions is denoted by the symbol $T_e(\mathbf{m})$. An enabled transition $t \in T_e(\mathbf{m})$ can *fire*, which changes the marking $\mathbf{m}$ to $\hat{\mathbf{m}}$ according to the equation

$$\hat{\mathbf{m}}(p) = \mathbf{m}(p) - \mathrm{card}(p^\bullet \cap \{t\}) + \mathrm{card}({}^\bullet p \cap \{t\}) \quad (1)$$

where the symbol $\mathrm{card}(\bullet)$ is used to denote the cardinality of the set argument, and $x^\bullet := \{y | (x, y) \in \Phi\}$. This notation is also applied to denote the predecessor or successor set of a set of places or transitions.

A string of transitions $\sigma = t_{j_1} t_{j_2} \cdots t_{j_k}$, where $t_{j_i} \in T (i \in \{1, 2, \cdots, k\})$ is said to be a *valid firing sequence* starting from the marking $\mathbf{m}$, if:

- the transition t_{j_1} is enabled under the marking $\mathbf{m}$;
- for $i \in \{1, 2, \cdots, k-1\}$ the firing of the transition t_{j_i} produces a marking under which the transition $t_{j_{i+1}}$ is enabled.

Given an initial marking $\mathbf{m}^0$, the set of *reachable markings* for $\mathbf{m}^0$ denoted by $\Re(N, \mathbf{m}^0)$ is the set of markings generated by all valid firing sequences starting with marking $\mathbf{m}^0$ in the PN N. At a marking $\mathbf{m}^1$, if the firing of a valid firing sequence σ results in a marking $\mathbf{m}^2$, we represent it as $\mathbf{m}^1 \rightarrow \sigma \rightarrow \mathbf{m}^2$. A transition $t \in T$ is *live* if

$$\forall \mathbf{m}^1 \in \Re(N, \mathbf{m}^0), \exists \mathbf{m}^2 \in \Re(N, \mathbf{m}^1) \text{ such that } t \in T_e(\mathbf{m}^2).$$

The PN shown in Fig. 1 is not live. This is because the marking $\mathbf{m} = [0\ 1\ 0\ 0\ 1]^T$ is reachable from the initial marking $\mathbf{m}^0 = [1\ 0\ 0\ 0\ 1]^T$, and $T_e(\mathbf{m}) = \emptyset$.

In the context of the marking being represented as a nonnegative, integral vector, the i, jth entry of the $n \times m$ *incidence matrix* $\mathbf{C}$ of the PN N is a matrix defined as

$$\mathbf{C}_{i,j} = \begin{cases} -1 & \text{if } p_i \in {}^\bullet t - t^\bullet, \\ 1 & \text{if } p_i \in t^\bullet - {}^\bullet t, \\ 0 & \text{otherwise.} \end{cases}$$

[1] In this paper we restrict our attention to *ordinary PN's*. This is implicitly assumed when we suppose $\Phi \subseteq (\Pi \times T) \cup (T \times \Pi)$.

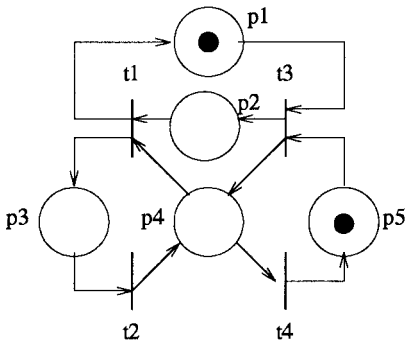

Fig. 1. A PN that is not live.

So, if $\mathbf{x}(\sigma)$ is the Parikh mapping of any valid firing sequence $\sigma \in T^*$ starting at $\mathbf{m}^0$, the resulting marking $\mathbf{m}$ can be represented as

$$\mathbf{m} = \mathbf{m}^0 + \mathbf{C}\mathbf{x}(\sigma). \tag{2}$$

Given two integer-valued vectors $\mathbf{x}, \mathbf{y}$, we use the notation $\mathbf{x} > \mathbf{y} (\mathbf{x} \geq \mathbf{y})$ if each component of $\mathbf{x}$ is greater than (greater than or equal to) the corresponding component of $\mathbf{y}$.

Given two PN's $N_i = (\Pi_i, T_i, \Phi_i, \mathbf{m}_i^0)(i = 1, 2)$ the *inclusion problem* for reachable sets involves the following question: $\Re(N_1, \mathbf{m}_1^0) \subseteq \Re(N_2, \mathbf{m}_2^0)$? This problem is undecidable (cf. [14, Th. 5.11] and [17, Th. 6.3]).

A *directed graph* $G = (V, A, \Psi)$ is an ordered 3-tuple, where V is a finite set of *vertices*, A is a finite set of *arcs*, and Ψ: $A \rightarrow V \times V$ is the *incidence function*. For each $a \in A$, if $\Psi(a) = (v_i, v_j)$, then the arc a is said to *originate* (*terminate*) at $v_i (v_j)$. Following the notation used for PN's, we define ${}^\bullet a = \{v_i\}$ and $a^\bullet = \{v_j\}$. For each $v \in V$, we define ${}^\bullet v = \{a \in A | a^\bullet = \{v\}\}$ and $v^\bullet = \{a \in A | {}^\bullet a = \{v\}\}$. A *path* in a directed graph is a sequence of edges $\sigma = a_1 a_2 \cdots a_k$ such that $a_i^\bullet = {}^\bullet a_{i+1} (i \in \{1, 2, \cdots, k-1\})$. The path σ originates at ${}^\bullet a_1$ and terminates at $a_k^\bullet$. The fact that there is a path labeled σ from vertex v_i to v_j is represented as $v_i \rightarrow \sigma \rightarrow v_j$. For any vertex $v \in V$, the set of vertices connected to v, $\{\hat{v} \in V | \exists \sigma \in A^*$, such that $v \rightarrow \sigma \rightarrow \hat{v}\}$, can be identified by a *breadth-first search* algorithm (cf. [5, Sec. 23.2]) with v being the source-vertex.

For a given PN $N = (\Pi, T, \Phi, \mathbf{m}^0)$, the *Karp and Miller Tree* (KM-tree) of N, $G(N, \mathbf{m}^0) = (V, A, \Psi)$ is a directed graph. Each vertex v_i is associated with an extended marking $\mu(i) \in (\mathcal{N} \cup \infty)^n$, and each edge a in the KM-tree is associated with a transition $\Gamma(a) \in T$. The algorithm for the construction of the KM-tree $G(N, \mathbf{m}^0) = (V, A, \Psi)$ for a given PN N is presented below (cf. [14, Sec. 4.2.1]).

- The root vertex of G is v_0 and $\mu(v_0) = \mathbf{m}^0$.
- Let v_i be a vertex already in G with a label $\mu(v_i) \in (\mathcal{N} \cup \infty)^m$, then:
  - if the label of v_i is identical to the label of some vertex v_j already in G, then v_i has no children and is marked as a *duplicate* of v_j;
  - if no transitions are enabled for the extended marking $\mu(v_i)$, then v_i is said to be a *terminal* vertex;
  - $\forall t_j \in T$ enabled under the extended marking $\mu(v_i)$, create a new vertex v_k in G. Also create a directed arc a_l such that ${}^\bullet a_l = \{v_i\}$ and ${}^\bullet a_l = \{v_k\}$ and $\Gamma(a_l) = t_j$. The extended marking $\mu(v_k)$ is computed as follows: $\forall l \in \{1, 2, \cdots, n\}$:
    - if $\mu(v_i)_l = \infty$, then $\mu(v_k)_l = \infty$;
    - if $\exists v_m$ on the path from v_0 to v_k with associated marking $\mu(v_m)$ such that: 1) $\mu(v_m) \leq \mu(v_i) + \mathbf{C}\mathbf{1}_j$, where $\mathbf{1}_j$ is the unit-vector that corresponds to the firing of t_j; and 2) $\mu(v_m)_l < (\mu(v_i) + \mathbf{C}\mathbf{1}_j)_l$, then $\mu(v_k)_l = \infty$;
    - otherwise, $\mu(v_k)_l = (\mu(v_i) + \mathbf{C}\mathbf{1}_j)_l$.

The KM-tree of any PN is finite (cf. [14, Th. 4.1], [17, Th. 4.1]). The coverability graph, $\hat{G}(N, \mathbf{m}^0) = (\hat{V}, \hat{A}, \hat{\Psi})$, of a PN is essentially the KM-tree, where the duplicate vertices are merged as one. Since edges originating from any vertex are assigned distinct transitions, a coverability graph can be interpreted as a deterministic, finite-state automation that generates a language that is a subset of T^*. Therefore, paths in coverability graphs can be identified by the sequence of transitions that correspond to the labels of the edges.

A CtlPN is expressed as an ordered 7-tuple: $M = (\Pi, T_u, T_c, \Phi, \mathbf{m}^0, C, B)$, where $\Pi = \{p_1, p_2, \cdots, p_n\}$ is a set of n *state places*, $T_u = \{t_1^u, t_2^u, \cdots, t_p^u\}$ is a set of p *uncontrollable transitions*, $T_c = \{t_1^c, t_2^c, \cdots, t_q^c\}$ is a set of q *controllable transitions*, $\Phi \subseteq (\Pi \times T) \cup (T \times \Pi)$ is a set of *state-arcs*; $C = \{c_1, c_2, \cdots, c_q\}$[2] is the set of *control places*; $B = \{(c_i, t_i^c) | i = 1, 2, \cdots, q\}$, is the set of *control arcs*; $\mathbf{m}^0$: $\Pi \rightarrow \mathcal{N}$ is the *initial marking function* (or the *initial marking*), and $\mathcal{N}$ is the set of nonnegative integers. The CtlPN $M = (\Pi, T_u, T_c, \Phi, \mathbf{m}^0, C, B)$ contains the underlying PN $N = (\Pi, T_u \cup T_c, \Phi, \mathbf{m}^0)$, where $m = p + q = \text{card}(T_u \cup T_c)$. We say the CtlPN M is *bounded* when the underlying N is bounded. In graphical representations of CtlPN's, controllable (uncontrollable) transitions are represented by dark/filled (empty/unfilled) rectangles, and we do not explicitly represent the control places.

A *control* $\mathbf{u}$: $C \rightarrow \{0, 1\}$ assigns a token load of zero or one to each control place. With the added provision that uncontrollable transitions are always control-enabled, the control can also be interpreted as an m-dimensional[3] binary vector $\mathbf{u} \in \{0, 1\}^m$. It would help to view the control $\mathbf{u}$ as follows: if the ith component of $\mathbf{u}$, or $\mathbf{u}(c_i)$, is zero (one), then transition t_i is control-disabled (control-enabled). For a given marking $\mathbf{m}$ (control $\mathbf{u}$), a transition $t_i \in T$ is said to be state-enabled (control-enabled) if $t_i \in T_e(\mathbf{m})$ (if $\mathbf{u}(c_i) = 1$). A transition that is control-enabled and state-enabled can fire, resulting in the marking given by (1). A *supervisory policy* $\mathcal{P}$: $\mathcal{N}^n \rightarrow \{0, 1\}^m$ is a total map that assigns a control for each potentially reachable marking.

For a given CtlPN and supervisory policy $\mathcal{P}$, a string of transitions $\sigma = t_{j_1} t_{j_2} \cdots t_{j_k}$, where $t_{j_i} \in T (i \in \{1, 2, \cdots, k\})$, is said to be a *valid firing sequence under supervision* starting from the marking $\mathbf{m}$, if:

[2]Note that $\text{card}(C) = \text{card}(T_c) = q$.

[3]Note that $m = p + q = \text{card}(T_u \cup T_c)$.

- the transition t_{j_1} is state-enabled under the marking $\mathbf{m}$, $\mathcal{P}(\mathbf{m})_{j_1} = 1$;
- for $i \in \{1, 2, \cdots, k-1\}$ the firing of the transition t_{j_i} produces a marking $\hat{\mathbf{m}}$ under which the transition $t_{j_{i+1}}$ is state-enabled and $\mathcal{P}(\hat{\mathbf{m}})_{j_{i+1}} = 1$.

For a given supervisory policy $\mathcal{P}$, the set of *reachable markings under supervision* for a CtlPN M with initial marking $\mathbf{m}^0$, denoted by $\Re(M, \mathbf{m}^0, \mathcal{P})$, is the set of markings generated by all valid firing sequences under supervision starting with marking $\mathbf{m}^0$ in the CtlPN M. The KM-tree for a CtlPN is the KM-tree of its underlying PN. For the CtlPN M, a transition $t_{j_i} \in T$ is *live* under $\mathcal{P}$ if

$$\forall\, \mathbf{m}^1 \in \Re(M, \mathbf{m}^0, \mathcal{P}), \exists\, a\mathbf{m}^2 \in \Re(M, \mathbf{m}^1, \mathcal{P}) \text{ such that } t_{j_i} \in T_e(\mathbf{m}^2) \text{ and } \mathcal{P}(\mathbf{m}^2)_{j_i} = 1.$$

A supervisory policy $\mathcal{P}$ enforces liveness in a CtlPN M if all transitions in M are live under $\mathcal{P}$. If the underlying PN N in a CtlPN M is live, then the trivial supervisory policy $\mathcal{P}(\mathbf{m})_i = 1, \forall \mathbf{m} \in \mathcal{N}^n, \forall i \in \{1, 2, \cdots, m\}$ enforces liveness in the CtlPN M. Given two supervisory policies $\mathcal{P}_1$ and $\mathcal{P}_2$ that enforce liveness in a CtlPN $M = (\Pi, T_u, T_c, \Phi, \mathbf{m}^0, C, B)$, we say $\mathcal{P}_1$ is *less restrictive* than $\mathcal{P}_2$ if $\forall \mathbf{m} \in \Re(M, \mathbf{m}^0, \mathcal{P}_1) \cap \Re(M, \mathbf{m}^0, \mathcal{P}_2)$: 1) $\mathcal{P}_1(\mathbf{m}) \geq \mathcal{P}_2(\mathbf{m})$, componentwise and 2) $\exists i \in \{1, 2, \cdots, m\}$ such that $\mathcal{P}_1(\mathbf{m})_i > \mathcal{P}_2(\mathbf{m})_i$. A supervisory policy $\mathcal{P}$ that enforces liveness is said to be *minimally restrictive* if there does not exist a supervisory policy that is less restrictive than $\mathcal{P}$.

The *weakest liberal precondition* of a set of markings $\mathcal{M} \subseteq \mathcal{N}^n$, denoted by $\Omega(\mathcal{M})$, is defined as

$$\Omega(\mathcal{M}) = \{\mathbf{m}^1 \in \mathcal{N}^n | (\forall t_u \in T_u, \forall \mathbf{m}^2 \in \mathcal{N}^n, \text{ if } \mathbf{m}^1 \to t_u \to \mathbf{m}^2, \text{ then } \mathbf{m}^2 \in \mathcal{M}) \text{ or } (\forall t_u \in T_u, \nexists \mathbf{m}^2 \in \mathcal{N}^n, \text{ s.t. } \mathbf{m}^1 \to t_u \to \mathbf{m}^2)\}.$$

We say $\mathcal{M}$ is *control invariant* if $\mathcal{M} \cap \Omega(\mathcal{M}) = \mathcal{M}$.

We will have the occasion to use the notion of weakest liberal precondition, $\Omega(\tilde{V})$, of a set of vertices, $\tilde{V} \subseteq \hat{V}$, in the coverability graph, $\hat{G} = (\hat{V}, \hat{A}, \hat{\Psi})$, where

$$\Omega(\tilde{V}) = \{\hat{v} \in \hat{V} | (\forall a \in \hat{v}^\bullet, \text{ if } \Gamma(a) \in T_u, \text{ and } {}^\bullet a = \{\hat{v}\}, \text{ then } a^\bullet \in \mathcal{V}) \text{ or } (\forall a \in \hat{v}^\bullet, \Gamma(a) \notin T_u)\}.$$

As with a subset of markings, we say $\tilde{V}$ is *control invariant* if $\tilde{V} \cap \Omega(\tilde{V}) = \tilde{V}$.

In the remainder we consider necessary and sufficient conditions for the existence of a supervisory control policy that enforces liveness in an arbitrary CtlPN with a possibly nonempty uncontrollable transition set. First, we present a few motivating examples.

III. Motivation via Examples

Consider the underlying PN of the CtlPN M in Fig. 2, which is not live. Transitions t_3 and t_4 are controllable, while transitions t_1 and t_2 are uncontrollable. The supervisory policy shown in tabular form in the same figure enforces liveness

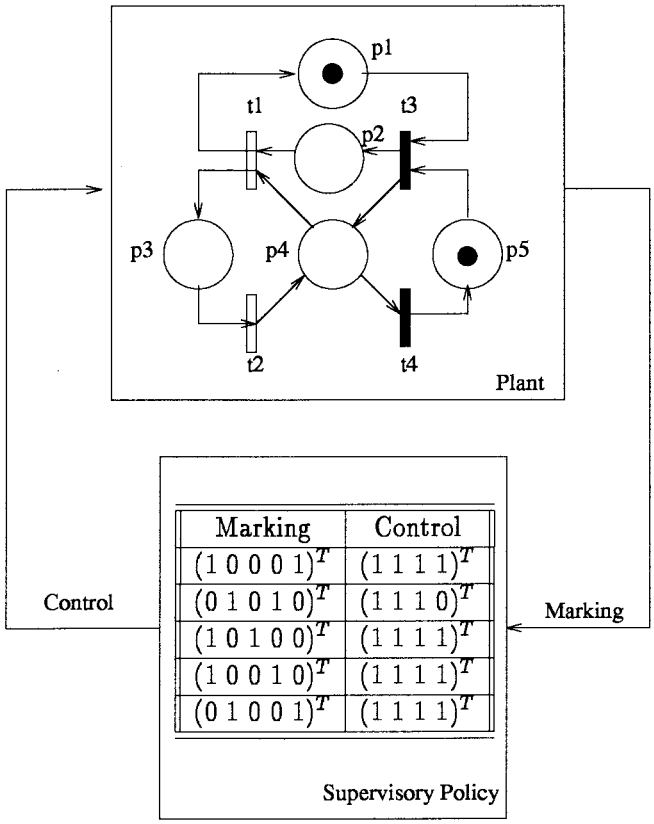

Fig. 2. A supervisory policy that enforces liveness in a plant CtlPN whose underlying PN is not live.

in the CtlPN. The marking $(0\ 1\ 0\ 0\ 1)^T$ is not reachable under supervision and the supervisory policy could have been undefined for this marking, but we chose to assign $\{1\}^4$ for the sake of completeness. It can also be shown that the supervisory policy shown here is minimally restrictive. The CtlPN in Fig. 2 is bounded; this permitted the expression of the supervisory policy as a table. When the plant CtlPN is unbounded, the supervisory policy is usually represented as a procedure that computes the control (i.e., the enabled/disabled transitions) for a given marking.

The CtlPN shown in Fig. 3 is not bounded. Also, its underlying PN is not live. In this paper, we use a C-like syntax as shown in the figure to describe supervisory policies that involve CtlPN's that are unbounded or have a large, finite set of reachable states. Under the supervisory policy, the transition t_1 is control-enabled only when the place set $\{p_6, p_7, p_8\}$ has a nonempty token-load. This supervisory policy enforces liveness.

For the CtlPN's shown in Fig. 4(a) and (b) it is easy to see that there can be no supervisory policy that enforces liveness. A less obvious example of a nonlive, unbounded CtlPN that cannot be made live via supervisory control is shown in Fig. 5. In this example, the first firing of transitions t_3 and t_8 will eventually result in a deadlock. These examples suggest that the condition for the existence of a supervisory policy that enforces liveness will concern the structure of the underlying PN. It is not hard to see that the initial marking will also influence the existence of a supervisory policy. As

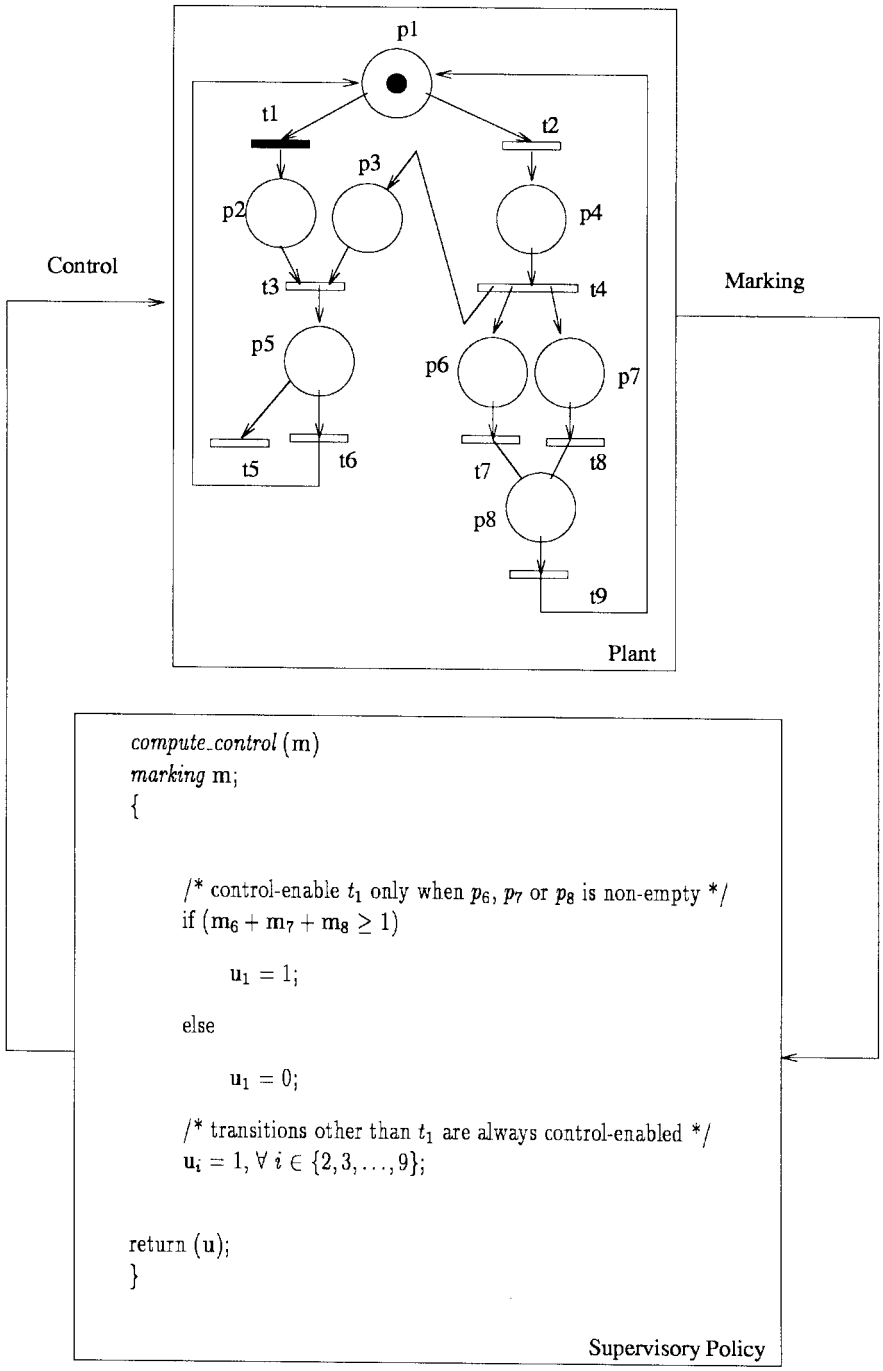

Fig. 3. A supervisory policy that enforces liveness in an unbounded plant CtlPN with an underlying PN that is not live.

an extreme example, consider the CtlPN of Fig. 2, with no tokens anywhere (i.e., the initial marking is the zero marking). Clearly, the supervisory policy can do nothing to enforce liveness. The existence of a supervisory policy that enforces liveness also depends on the set of uncontrollable transitions. To see this, consider the CtlPN shown in Fig. 2. Without supervision, the set of reachable markings of the underlying PN is $\{(1, 0, 0, 0, 1)^T, (0\ 1\ 0\ 1\ 0)^T, (1\ 0\ 1\ 0\ 0)^T, (1\ 0\ 0\ 1\ 0)^T, (0\ 1\ 0\ 0\ 1)^T\}$. For this CtlPN, a supervisory policy enforces liveness if and only if the set of markings reachable under supervision is $\{(1\ 0\ 0\ 0\ 1)^T, (0\ 1\ 0\ 1\ 0)^T, (1\ 0\ 1\ 0\ 0)^T, (1\ 0\ 0\ 1\ 0)^T\}$. Consequently, if t_4 is not controllable, it is not hard to see that there cannot be any supervisory policy that enforces liveness. We will see shortly that the existence of a supervisory policy that enforces liveness in a CtlPN depends on: 1) the structure of the underlying PN; 2) the initial marking; and 3) the set of uncontrollable transitions. In the next section we survey some relevant results in the literature.

IV. Review of Relevant Previous Work

In [8] Holloway and Krogh solve a class of forbidden marking (i.e., forbidden state) problems where the plant is represented as a marked-graph PN. For this problem they also consider in [9] the issue of guaranteeing closed-loop liveness of their supervisory policies. They assume the plant PN is

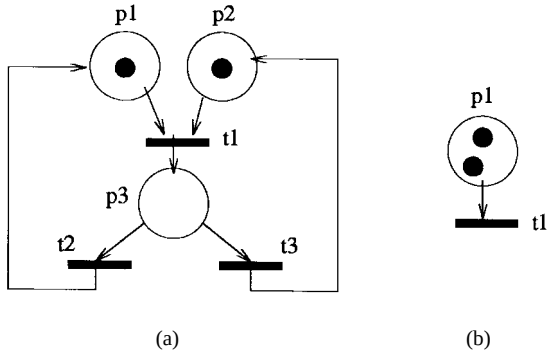

Fig. 4. Two CtlPN's that cannot be made live via supervisory control.

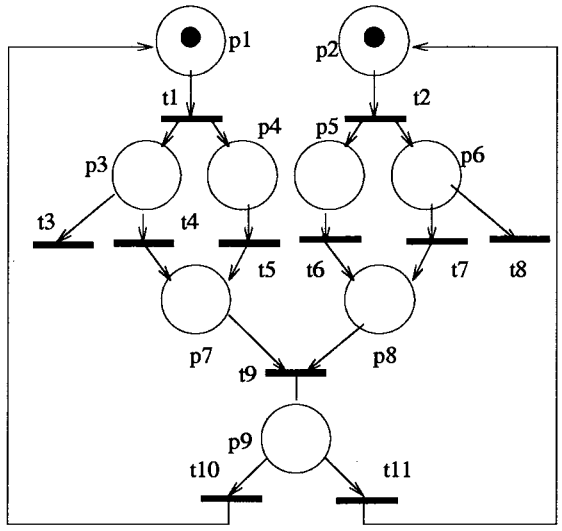

Fig. 5. Another CtlPN that cannot be made live via supervisory control.

live at the beginning. There is no way to enforce liveness via supervisory control in marked-graph PN's that are not live originally. This is because a marked-graph PN is not live if and only if some directed circuit is empty. Additionally, the number of tokens in a directed circuit of a marked-graph PN remains invariant. So, if a directed circuit is empty at the initial marking, it will remain empty for all reachable markings. In other words, lack of liveness in marked graph PN's is only due to a lack of sufficient tokens at initialization and not due to any "dynamics." Supervisory control can do nothing to rectify this situation.

In [16] Ramadge and Wonham consider plants with *marked states*. These states imply the successful completion of some task in the plant. In this context they introduce the notion of *nonblocking supervisors* that guarantee that the supervisory control actions do not result in a sequence of states that can never be completed to reach a marked state. Assuming transitions in a PN are assigned distinct event symbols, PN liveness specifications guarantee for each event, and for each reachable state, there is a valid event sequence containing the event under consideration. It is quite possible that the PN liveness specification can be converted into an equivalent specification that guarantees nonblocking for a particular choice of marked states. For liveness specifications, the choice of marked states is not apparent. This issue is revisited in Section VI.

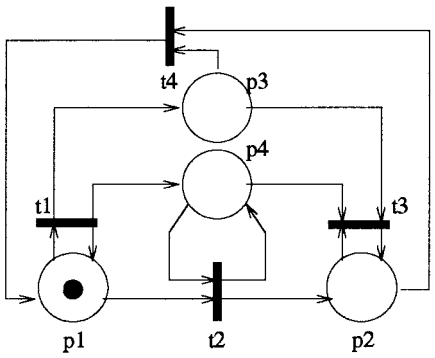

Fig. 6. An unbounded PN where an exhaustive path analysis of the coverability graph for liveness yields spurious results.

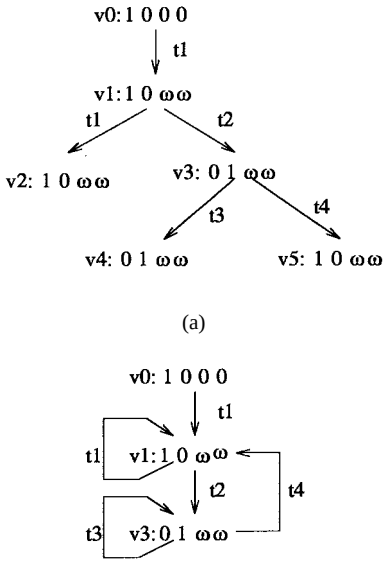

Fig. 7. (a) The KM-tree and (b) coverability graph of the PN in Fig. 6.

In [22] Vishwanadham *et al.* consider deadlock avoidance for PN's with finite reachability sets. Their approach essentially uses a search on the coverability graphs. To illustrate the shortcomings of exhaustive path analysis of the coverability graph for unbounded PN's, we consider the unbounded PN shown in Fig. 6. This PN is obtained from Peterson's text [14, Fig. 4.23], with appropriate modifications to yield arcs with unity weights. This unbounded PN would deadlock after the execution of the firing sequence $t_1t_2t_3$. However, the coverability graph of this unbounded PN, shown in Fig. 7(b), does not suggest the existence of the deadlock. Moreover, an analysis of the paths in the coverability graph yields the conclusion that the unbounded PN is live, while in reality it is not. This should come as no surprise as the coverability graph has limited use when the PN is unbounded.

The *limited lookahead* heuristic is frequently used in this context. Its performance is well understood for the forbidden state problem in DEDS [4]. Reference [22] contains an application of this heuristic to a specific PN model of a simple manufacturing system. However, for enforcing liveness, or

deadlock avoidance in PN's, there are no explicit directives on the required search depth that can guarantee liveness for arbitrary PN models. Banaszak and Krogh [1] consider PN models of concurrent process flows that involve resource sharing. The supervisory policy in their case is a restriction on the set of enabled transitions that can fire at any instant. They present a provably correct deadlock avoidance algorithm for this class of PN's.

The list of references surveyed herein is certainly not complete. In particular, the issue of deadlock avoidance has also been extensively dealt with in the operating systems literature. The widely held opinion is that algorithms that prevent the occurrence of a deadlock by negating one of the four conditions necessary for deadlocks are often too conservative. The common consensus is that the issue of deadlock avoidance is still open, despite significant advances over the past two decades (cf. [20, p. 134]).

V. Main Results

We first present a necessary and sufficient condition for the existence of a supervisory policy that enforces liveness in Theorem 5.1. This condition, stated on the potentially unbounded set of reachable markings, is shown to be testable when the set of uncontrollable transitions is empty (cf. Corollary 5.1) and untestable when the set of uncontrollable transitions is nonempty (cf. Corollary 5.2). In Corollary 5.3 we show the corresponding problem is solvable for bounded CtlPN's.

Theorem 5.1: For a given CtlPN $M = (\Pi, T_u, T_c, \Phi, \mathbf{m}^0, C, B)$, with an underlying PN $N = (\Pi, T_u \cup T_c, \Phi, \mathbf{m}^0)$, there exists a supervisory policy $\mathcal{P}{:}\mathcal{N}^n \rightarrow \{0,1\}^m (m = \text{card}(T_u \cup T_c))$ that enforces liveness, if and only if $\exists$ a control invariant subset of markings $\mathcal{M} \subseteq \Re(N, \mathbf{m}^0)$, such that:

1) $\forall \mathbf{m}^1 \in \mathcal{M}, \exists \mathbf{m}^2, \mathbf{m}^3 \in \mathcal{M}, \exists$ a valid firing sequence $\sigma = \sigma_1\sigma_2$ in N starting from $\mathbf{m}^1$ such that $\mathbf{m}^1 \rightarrow \sigma_1 \rightarrow \mathbf{m}^2 \rightarrow \sigma_2 \rightarrow \mathbf{m}^3$, and:
   a) $\mathbf{m}^3 \geq \mathbf{m}^2$;
   b) the Parikh mapping of σ_2, $\mathbf{x}(\sigma_2)$, satisfies the requirement $\mathbf{x}(\sigma_2) > 0$ (i.e., all transitions appear in σ_2 at least once);
   c) $\forall \sigma_3 \in pr(\sigma_1\sigma_2), \mathbf{m}^1 \rightarrow \sigma_3 \rightarrow \mathbf{m}^4 \Rightarrow \mathbf{m}^4 \in \mathcal{M}$, where $pr(\bullet)$ is the prefix-set of the string argument.
2) $\mathbf{m}^0 \in \mathcal{M}$.

Proof (Only If Part): Let $\mathcal{P}$ be a supervisory policy that enforces liveness. It follows that $\forall t_i \in T$

$$\forall \mathbf{m}^1 \in \Re(M, \mathbf{m}^0, \mathcal{P}), \exists \hat{\mathbf{m}} \in \Re(M, \mathbf{m}^1, \mathcal{P}) \text{ such that } t_i \in T_e(\hat{\mathbf{m}}) \text{ and } \mathcal{P}(\hat{\mathbf{m}})_i = 1. \quad (3)$$

We show that $\Re(M, \mathbf{m}^0, \mathcal{P})$ satisfies the requirements of the theorem. $\Re(M, \mathbf{m}^0, \mathcal{P})$ is control invariant, and $\Re(M, \mathbf{m}^0, \mathcal{P}) \subseteq \Re(N, \mathbf{m}^0)$, by definition. Also, $\mathbf{m}^0 \in \Re(M, \mathbf{m}^0, \mathcal{P})$. For each $\mathbf{m}^1 \in \Re(M, \mathbf{m}^0, \mathcal{P})$, (3) suggests the existence of an infinite sequence of markings $\{\hat{\mathbf{m}}^i\}_{i=1}^{\infty}$ and firing sequences $\{\sigma_i\}_{i=1}^{\infty}$, such that

$$\mathbf{m}^1 \rightarrow \sigma_1 \rightarrow \hat{\mathbf{m}}^1 \rightarrow \sigma_2 \rightarrow \hat{\mathbf{m}}^2 \rightarrow \sigma_3 \rightarrow \cdots$$

where the Parikh mappings of each σ_i, $\mathbf{x}(\sigma_i)$ satisfies the requirement $\mathbf{x}(\sigma_i) > 0$. The infinite sequence of markings $\{\hat{\mathbf{m}}^i\}_{i=1}^{\infty}$ should contain an infinite nondecreasing subsequence $\{\mathbf{m}^i\}_{i=1}^{\infty}$ (cf. [14, Lemma 4.3], for a formal proof[4]). That is, elements of the sequence $\{\mathbf{m}^i\}_{i=1}^{\infty}$ satisfy the requirement $\mathbf{m}^1 \geq \mathbf{m}^2 \geq \cdots$. So, $\exists \sigma_1, \sigma_2$, such that $\mathbf{m}^1 \rightarrow \sigma_1 \rightarrow \mathbf{m}^2 \rightarrow \sigma_2 \rightarrow \mathbf{m}^3$, where $\mathbf{m}^3 \geq \mathbf{m}^2$ and $\mathbf{x}(\sigma_2) > 0$. Also, $\forall \sigma_3 \in pr(\sigma_1\sigma_2), \mathbf{m}^1 \rightarrow \sigma_3 \rightarrow \mathbf{m}^4 \Rightarrow \mathbf{m}^4 \in \Re(M, \mathbf{m}^0, \mathcal{P})$.

(If Part): For this part we present a supervisory policy that enforces liveness given that $\exists \mathcal{M} \subseteq \Re(N, \mathbf{m}^0)$ that satisfies the requirements of the theorem. Let $\mathcal{P}{:}\ \Re(N, \mathbf{m}^0) \rightarrow \{0,1\}^m$ be a supervisory policy defined by

$$\mathcal{P}(\mathbf{m})_i = \begin{cases} 0, & \text{if } \mathbf{m} \rightarrow t_i \rightarrow \hat{\mathbf{m}} \text{ and } \hat{\mathbf{m}} \notin \mathcal{M} \\ 1, & \text{otherwise.} \end{cases}$$

Since $\mathcal{M}$ is control invariant, if $\mathbf{m} \in \mathcal{M}$ and $\exists t_u \in T_u$, such that $\mathbf{m} \rightarrow t_u \rightarrow \hat{\mathbf{m}}$, then $\hat{\mathbf{m}} \in \mathcal{M}$. Note that $\forall \mathbf{m}^1 \in \Re(M, \mathbf{m}^0, \mathcal{P}), \exists \sigma_1, \sigma_2$, such that $\mathbf{m}^1 \rightarrow \sigma_1 \rightarrow \mathbf{m}^2 \rightarrow \sigma_2 \rightarrow \mathbf{m}^3$, where $\mathbf{m}^3 \geq \mathbf{m}^2$ and the Parikh mapping of σ_2, $\mathbf{x}(\sigma_2)$ satisfies the requirement $\mathbf{x}(\sigma_2) > 0$. Using induction over the length of any valid firing sequence under supervision, the firing sequence $\sigma_1\sigma_2$ can be shown to be valid under supervision starting from $\mathbf{m}^1$ as $\forall\ \sigma_3 \in pr(\sigma_1\sigma_2), \mathbf{m}^1 \rightarrow \sigma_3 \rightarrow \mathbf{m}^4 \Rightarrow \mathbf{m}^4 \in \mathcal{M}$. This establishes the fact that $\mathcal{P}$ enforces liveness. □

Theorem 5.1 presented a necessary and sufficient condition for the existence of a supervisory policy that enforces liveness in an arbitrary CtlPN, but this condition is stated over a potentially unbounded set of reachable markings. For CtlPN's with an empty set of uncontrollable transitions (i.e., $T_u = \emptyset$), any subset of reachable markings is trivially control invariant. Therefore, for this class of CtlPN's, a supervisory policy that enforces liveness exists if and only if there are two markings $\mathbf{m}^1, \mathbf{m}^2$ and a string of transitions $\sigma_1\sigma_2$ such that:

1) $\mathbf{m}^0 \rightarrow \sigma_1 \rightarrow \mathbf{m}^1 \rightarrow \sigma_2 \rightarrow \mathbf{m}^2$;
2) $\mathbf{m}^2 \geq \mathbf{m}^1$;
3) $\mathbf{x}(\sigma_2) > 0$ (i.e., every transition appears at least once in σ_2).

In Theorem 5.2 we show the above requirement can be effectively converted into a test on circuits in the finite coverability graph of the underlying PN. The Appendix contains an algorithm that tests the condition of Theorem 5.2. Therefore, for CtlPN's with an empty uncontrollable transition set the existence of a supervisory policy that enforces liveness is decidable. This is formally stated in Corollary 5.1.

We now introduce some additional notation that will simplify the following discussion. For a vertex v_j in the coverability graph we define the set $I_{v_j}^{\infty}$ of indexes of places assigned an infinite-token load under the extended marking $\mu(v_j)$ as

$$I_{v_j}^{\infty} = \{\alpha \in \{1, 2, \cdots, n\} | (\mu(v_j))_\alpha = \infty\}.$$

In Theorem 5.2 we show for a given marking $\mathbf{m}^0$, there exists a valid firing sequence $\sigma_1\sigma_2$ in the underlying PN, such that $\mathbf{m}^0 \rightarrow \sigma_1 \rightarrow \mathbf{m}^1 \rightarrow \sigma_2 \rightarrow \mathbf{m}^2$, and $\mathbf{m}^2 \geq \mathbf{m}^1$, and $\mathbf{x}(\sigma_2) > 0$, if and only if there is a path $v_0 \rightarrow \hat{\sigma}_1 \rightarrow v_1 \rightarrow \hat{\sigma}_2 \rightarrow v_1$, in the coverability graph $\hat{G}(N, \mathbf{m}^0)$, such that $\mathbf{x}(\hat{\sigma}_2) > 0$

[4]This result is established for the extended nonnegative integral vectors, but the proof applies to nonnegative integral vectors as well.

and $\mathbf{Cx}(\hat{\sigma}_2) \geq 0$. In words, we convert the condition on the (potentially unbounded) reachable markings of Theorem 5.1 to an equivalent test on circuits in the (finite) coverability graph.

Theorem 5.2: For a CtlPN $M = (\Pi, T_u, T_c, \Phi, \mathbf{m}^0, C, B)$ with an underlying PN $N = (\Pi, T, \Phi, \mathbf{m}^0)$, there exists a valid firing sequence $\sigma = \sigma_1\sigma_2$, in N, starting from $\mathbf{m}^0$, such that:

1) $\mathbf{m}^2 \geq \mathbf{m}^1$;
2) the Parikh mapping of σ_2, $\mathbf{x}(\sigma_2)$ satisfies the requirement $\mathbf{x}(\sigma_2) > 0$, where $\mathbf{m}^0 \rightarrow \sigma_1 \rightarrow \mathbf{m}^1 \rightarrow \sigma_2 \rightarrow \mathbf{m}^2$ if and only if there exists a path $\hat{\sigma} = \hat{\sigma}_1\hat{\sigma}_2$ in the coverability graph $\hat{G}(N, \mathbf{m}^0)$, such that $v_0 \rightarrow \hat{\sigma}_1 \rightarrow v_1 \rightarrow \hat{\sigma}_2 \rightarrow v_1$, and the Parikh mapping of $\hat{\sigma}_2$, $\mathbf{x}(\hat{\sigma}_2)$, satisfies the requirement $\mathbf{x}(\hat{\sigma}_2) > 0$ and $\mathbf{Cx}(\hat{\sigma}_2) \geq 0$.

Proof (Only If Part): From the given conditions it can be inferred that $\sigma_1\sigma_2\sigma_2$ is a valid firing sequence. Let $\mathbf{m}^0 \rightarrow \sigma_1 \rightarrow \mathbf{m}^1 \rightarrow \sigma_2 \rightarrow \mathbf{m}^2 \rightarrow \sigma_2 \rightarrow \mathbf{m}^3$. From [17, Lemma 4.6], there exists a path in $\hat{G}(N, \mathbf{m}^1)$ that corresponds to the valid firing sequence $\sigma_1\sigma_2\sigma_2$. Let $v_0 \rightarrow \sigma_1 \rightarrow v_1 \rightarrow \sigma_2 \rightarrow v_2 \rightarrow \sigma_2 \rightarrow v_3$ in $\hat{G}(N, \mathbf{m}^1)$.

From the construction of the KM-tree and the subsequent construction of the coverability graph, we infer that $I^\infty_{v_1} \subseteq I^\infty_{v_2} \subseteq I^\infty_{v_3}$ (cf. [17, Observation 1, Sec. 4.2]). Since $\mu(v_0) = \mathbf{m}^0$, we infer $(\mu(v_1))_k = (\mathbf{m}^1)_k, k \notin I^\infty_{v_1}$; $(\mu(v_2))_k = (\mathbf{m}^2)_k k \notin I^\infty_{v_2}$; and $(\mu(v_3))_k = (\mathbf{m}^3)_k k \notin I^\infty_{v_3}$ (cf. [17, Observation 4, Sec. 4.2]). Since $I^\infty_{v_1} \subseteq I^\infty_{v_2}$, we infer $(\mu(v_2))_k \geq (\mu(v_1))_k, k \in I^\infty_{v_2}$. Since $\mathbf{m}^2 \geq \mathbf{m}^1$, we infer $(\mu(v_2))_k \geq (\mu(v_1))_k, k \notin I^\infty_{v_2}$. Therefore, $\mu(v_2) \geq \mu(v_1)$, if $\mathbf{m}^3_k > \mathbf{m}^2_k \Rightarrow (\mathbf{Cx}(\sigma_2))_k > 0 \Rightarrow \mathbf{m}^2_k > \mathbf{m}^1_k \Rightarrow k \in I_{v_2}$. But $I_{v_2} \subseteq I_{v_3}$, therefore $I_{v_3} = I_{v_2}$. So, $(\mu(v_3))_k = (\mu(v_2))_k = \infty, k \in I^\infty_{v_2} = I^\infty_{v_3}$. For $k \notin I^\infty_{v_2} = I^\infty_{v_3}, (\mu(v_3))_k = \mathbf{m}^3_k = \mathbf{m}^2_k + (\mathbf{Cx}(\sigma_2))_k = \mathbf{m}^2_k = (\mu(v_2))_k$. Therefore, $\mu(v_3) = \mu(v_2) \Rightarrow v_3 = v_2$. Letting $\hat{\sigma}_1 = \sigma_1\sigma_2, \hat{\sigma}_2 = \sigma_2$, we satisfy every required condition.

(If Part): The critical observation in this part of the proof is that if the token loads of the places whose indexes are in $I^\infty_{v_2}$ are made arbitrarily large for some marking reachable from $\mathbf{m}^0$, following this, $\hat{\sigma}_2$ can be repeated *ad infinitum*. From [17, Th. 4.2], we know that $\forall k \in \mathcal{N}, \exists \mathbf{m}^1 \in \Re(N, \mathbf{m}^0)$, such that: 1) $(\mathbf{m}^1)_i \geq k, i \in I^\infty_{v_2}$ and 2) $(\mathbf{m}^1)_i = (\mu(v_2))_i < \infty, i \notin I^\infty_{v_2}$. By choosing k large enough, we guarantee the validity of the firing sequence $\hat{\sigma}_2$ at $\mathbf{m}^2$. Let $\mathbf{m}^0 \rightarrow \tilde{\sigma} \rightarrow \mathbf{m}^1 \rightarrow \hat{\sigma}_2 \rightarrow \mathbf{m}^2$. Since $\mathbf{Cx}(\hat{\sigma}_2) \geq 0$, it follows: $\mathbf{m}^2 \geq \mathbf{m}^1$. Also, we are given the fact that $\mathbf{x}(\hat{\sigma}_2) > 0$, hence the result. □

Corollary 5.1: The existence of a supervisory policy that enforces liveness in a CtlPN $M = (\Pi, \emptyset, T_c, \Phi, \mathbf{m}^0, C, B)$ (i.e., the set of uncontrollable transitions in M is empty) is decidable.

Corollary 5.1 follows directly from Theorems 5.1 and 5.2, the procedure outlined in the Appendix, and the fact that the set of uncontrollable transitions is empty. When the set of uncontrollable transitions is nonempty, the above requirement is necessary but not sufficient for the existence of a supervisory policy as the sequence of markings reached from $\mathbf{m}^0$ by the firing of $\sigma_1\sigma_2$ might not be control invariant. Theorem 5.2 provides no insight into this requirement. This leads us to the existence of a supervisory policy that enforces liveness when the set of uncontrollable transitions in a CtlPN is nonempty. In Corollary 5.2 we show this problem is unsolvable. Toward this end we construct a CtlPN $\tilde{M}$ from two arbitrary PN's $N_i = (\Pi_i, T_i, \Phi_i, \mathbf{m}^0_i)(i = 1, 2)$ and show that there exists a supervisory policy that enforces liveness in $\tilde{M}$ if and only if $\Re(N_1, \mathbf{m}^0_1) \subseteq \Re(N_2, \mathbf{m}^0_2)$. Since the inclusion problem for reachable sets (i.e., the problem $\Re(N_1, \mathbf{m}^0_1) \subseteq \Re(N_2, \mathbf{m}^0_2)$?) is undecidable (cf. [14, Th. 5.11] and [17, Th. 6.3]) it follows that the existence of a supervisory policy that enforces liveness for an arbitrary CtlPN is also undecidable.

We first present the details of the construction of $\tilde{M}$ from two arbitrary PN's $N_i = (\Pi_i, T_i, \Phi_i, \mathbf{m}^0_i)(i = 1, 2)$. Without loss in generality let $\Pi_i = \{p^i_1, p^i_2, \cdots, p^i_n\}(i = 1, 2)$. That is, $\mathrm{card}(\Pi_1) = \mathrm{card}(\Pi_2) = n$. For the sake of completeness, let $T_i = \{t^i_1, t^i_2, \cdots, t^i_{m_i}\}(i = 1, 2)$. We first construct a new PN $\tilde{N} = (\tilde{\Pi}, \tilde{T}, \tilde{\Phi}, \tilde{\mathbf{m}}^0)$ as follows.

- $\tilde{\Pi} \leftarrow \Pi_1 \cup \Pi_2$.
- $\tilde{T} \leftarrow T_1 \cup T_2$.
- $\tilde{\Phi} \leftarrow \Phi_1 \cup \Phi_2$.
- Create $n+4$ new and unused places $\{\pi_1, \pi_2, \cdots, \pi_{n+4}\}$ (i.e., $\{\pi_1, \pi_2, \cdots, \pi_{n+4}\} \cap \tilde{\Pi} = \emptyset$) and modify $\tilde{\Pi}$ as $\tilde{\Pi} \leftarrow \tilde{\Pi} \cup \{\pi_1, \pi_2, \cdots, \pi_{n+4}\}$.
- Create n new and unused places $\{\hat{\pi}_3, \hat{\pi}_4, \cdots, \hat{\pi}_{n+2}\}$ (i.e., $\{\hat{\pi}_3, \hat{\pi}_4, \cdots, \hat{\pi}_{n+2}\} \cap \tilde{\Pi} = \emptyset$) and modify $\tilde{\Pi}$ as $\tilde{\Pi} \leftarrow \tilde{\Pi} \cup \{\hat{\pi}_3, \hat{\pi}_4, \cdots, \hat{\pi}_{n+2}\}$.
- Create $n + 4$ new and unused transitions $\{\tau_1, \tau_2, \cdots, \tau_{n+4}\}$ (i.e., $\{\tau_1, \tau_2, \cdots, \tau_{n+4}\} \cap \tilde{T} = \emptyset$) and modify $\tilde{T}$ as $\tilde{T} \leftarrow \tilde{T} \cup \{\tau_1, \tau_2, \cdots, \tau_{n+4}\}$.
- Create $2n$ new and unused transitions $\{\hat{\tau}^i_j | i \in \{1, 2\}, j \in \{1, 2, \cdots, n\}\}$ (i.e., $\{\hat{\tau}^i_j | i \in \{1, 2\}, j \in \{1, 2, \cdots, n\}\} \cap \tilde{T} = \emptyset$) and modify $\tilde{T}$ as $\tilde{T} \leftarrow \tilde{T} \cup \{\hat{\tau}^i_j | i \in \{1, 2\}, j \in \{1, 2, \cdots, n\}\}$.
- Create $2n$ new and unused transitions $\{\tilde{\tau}^i_j | i \in \{1, 2\}, j \in \{1, 2, \cdots, n\}\}$ (i.e., $\{\tilde{\tau}^i_j | i \in \{1, 2\}, j \in \{1, 2, \cdots, n\}\} \cap \tilde{T} = \emptyset$) and modify $\tilde{T}$ as $\tilde{T} \leftarrow \tilde{T} \cup \{\tilde{\tau}^i_j | i \in \{1, 2\}, j \in \{1, 2, \cdots, n\}\}$.
- $\forall t \in T_1$, modify $\tilde{\Phi}$ as $\tilde{\Phi} \leftarrow \tilde{\Phi} \cup \{(\pi_1, t), (t, \pi_1)\}$ (i.e., π_1 self-loops on all transitions of $\tilde{N}$ that originally belonged to N_1).
- $\forall t \in T_2$, modify $\tilde{\Phi}$ as $\tilde{\Phi} \leftarrow \tilde{\Phi} \cup \{(\pi_2, t), (t, \pi_2)\}$ (i.e., π_2 self-loops on all transitions of $\tilde{N}$ that originally belonged to N_2).
- $\forall i \in \{1, 2, \cdots, n+3\}$, modify $\tilde{\Phi}$ as $\tilde{\Phi} \leftarrow \tilde{\Phi} \cup \{(\pi_i, \tau_i), (\tau_i, \pi_{i+1})\}$.
- Modify $\tilde{\Phi}$ as $\tilde{\Phi} \leftarrow \tilde{\Phi} \cup \{(\pi_{n+4}, \tau_{n+4})\}$.
- $\forall p \in \tilde{\Pi}$, modify $\tilde{\Phi}$ as $\tilde{\Phi} \leftarrow \tilde{\Phi} \cup \{(\tau_{n+4}, p)\}$ (i.e., every place of $\tilde{N}$ is an output of τ_{n+4}).
- $\forall i \in \{1, 2, \cdots, n\}$, modify $\tilde{\Phi}$ as $\tilde{\Phi} \leftarrow \tilde{\Phi} \cup \{(\pi_{i+2}, \hat{\tau}^1_i), (p^1_i, \hat{\tau}^1_i), (\hat{\tau}^1_i, \hat{\pi}_{i+3}), (\hat{\pi}_{i+3}, \hat{\tau}^2_i), (p^2_i, \hat{\tau}^2_i), (\hat{\tau}^2_i, \pi_{i+2})\}$.
- $\forall i \in \{1, 2\}, \forall j \in \{1, 2, \cdots, n\}$, modify $\tilde{\Phi}$ as $\tilde{\Phi} \leftarrow \tilde{\Phi} \cup \{(\pi_{n+3}, \tilde{\tau}^i_j), (p^i_j, \tilde{\tau}^i_j)\}$.
- $\forall i \in \{1, 2\}, \forall p \in \Pi_i, \tilde{\mathbf{m}}^0(p) = \mathbf{m}^0_i(p)$.
- $\tilde{\mathbf{m}}^0(\pi_1) = 1$.
- $\forall p \in \tilde{\Pi} - \{\Pi_1 \cup \Pi_2 \cup \{\pi_1\}\}, \tilde{\mathbf{m}}^0(p) = 0$.

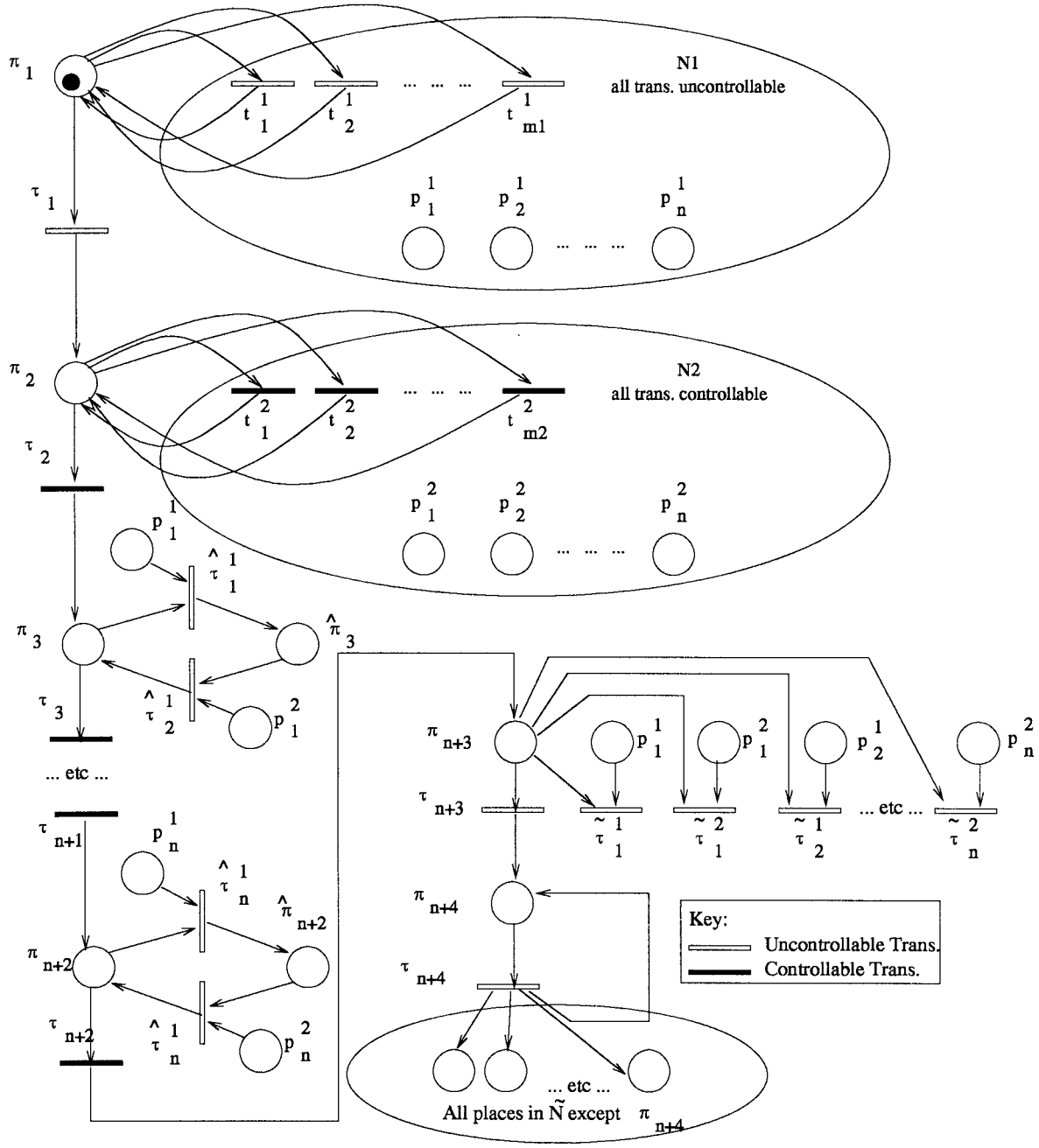

Fig. 8. A graphical illustration of the construction of the CtlPN $\tilde{M}$ from two arbitrary PN's, N_1 and N_2.

From $\tilde{N} = (\tilde{\Pi}, \tilde{T}, \tilde{\Phi}, \tilde{\mathbf{m}}^0)$ create a CtlPN $\tilde{M} = (\tilde{\Pi}, \tilde{T}_u, \tilde{T}_c, \tilde{\Phi}, \tilde{\mathbf{m}}^0, C, B)$ as follows.

- $\tilde{T}_c = T_2 \cup \{\tau_2, \tau_3, \cdots, \tau_{n+2}\}$.
- $\tilde{T}_u = \tilde{T} - \tilde{T}_c$.
- The set of control places C is defined by the following properties: 1) $\forall t_i \in T_c, \exists c_i \in C$; and 2) $\text{card}(C) = \text{card}(T_c)$.
- The set of control arcs B is defined as $B = \{(c_i, t_i) | t_i \in T_c\}$.

Fig. 8 shows the CtlPN resulting from the above construction.

Theorem 5.3: The CtlPN $\tilde{M} = (\tilde{\Pi}, \tilde{T}_u, \tilde{T}_c, \tilde{\Phi}, \tilde{\mathbf{m}}^0, C, B)$ constructed from two arbitrary PN's $N_i = (\Pi_i, T_i, \Phi_i, \mathbf{m}_i^0)(i = 1, 2)$ as shown above, has a supervisory policy that enforces liveness if and only if $\Re(N_1, \mathbf{m}_1^0) \subseteq \Re(N_2, \mathbf{m}_2^0)$.

Proof: From the construction, we observe that a supervisory policy enforces liveness if and only if, under supervision, a marking that places a token in π_{n+4} is reachable from any other reachable marking.

The sufficiency of the above requirement follows from the fact that $\tau_{n+4}^{\bullet} = \tilde{\Pi}$ and ${}^{\bullet}\tau_{n+4} = \{\pi_{n+4}\}$. To notice the necessity of the above condition we note the liveness of τ_1 can only be guaranteed by the repeated replenishment of the token-load of π_1. In $\tilde{M}$ this can only be achieved by the repeated firing of τ_{n+4}.

Since all transitions in T_1 and τ_1 are uncontrollable, the restriction of the marking $\hat{\mathbf{m}}$ of $\tilde{M}$, resulting from the first firing of τ_1 to those places originally in Π_1, can be any member of $\Re(N_1, \mathbf{m}_1^0)$. Following the first firing of the uncontrollable transition τ_1, there exists a valid firing sequence involving only the controllable transitions in T_2, resulting in a marking $\tilde{\mathbf{m}}$ of $\tilde{M}$ such that $\forall\, j \in \{1, 2, \cdots, n\}$, $\tilde{\mathbf{m}}(p_j^1)(= \hat{\mathbf{m}}(p_j^1)) = \tilde{\mathbf{m}}(p_i^2)$ if and only if $\Re(N_1, \mathbf{m}_1^0) \subseteq \Re(N_2, \mathbf{m}_2^0)$.

The single token in π_2 following the first firing of τ_1 can be guaranteed to be delivered to π_{n+3} if and only if for the marking resulting from the first firing of the controllable transition τ_2, the number of tokens in p_i^2, is at least as much as the number of tokens in $p_i^1 (i \in \{1, 2, \cdots, n\})$. To see this, note that if $\exists\, i \in \{1, 2, \cdots, n\}$, such that the number of tokens in p_i^2 is less than the number of tokens in p_i^1, then when the single token in π_2 following the first firing of τ_1 is at π_{i+2}, there exists an uncontrollable sequence of transitions that will result in a frozen token in $\hat{\pi}_{i+2}$.

Additionally, if any of the places in $\Pi_1 = \{p_1^1, p_2^1, \cdots, p_n^1\}$ or $\Pi_2 = \{p_1^2, p_2^2, \cdots, p_n^2\}$ are nonempty when π_{n+3} has a token and π_{n+4} is empty, one of the $\tilde{\tau}_j^i$ transitions can fire uncontrollably, resulting in a marking reachable under supervision from which a marking that places a token in π_{n+4} can never be reached.

Therefore, any supervisory policy that enforces liveness must guarantee the emptiness of the places originally in Π_1 and Π_2 when π_{n+4} is empty and there is a token in π_{n+3}.

(Only If Part): Let $\Re(N_1, \mathbf{m}_1^0) \not\subseteq \Re(N_2, \mathbf{m}_2^0) \Rightarrow \exists \mathbf{m} \in \Re(N_1, \mathbf{m}_1^0) - \Re(N_2, \mathbf{m}_2^0)$. Since all transitions in $T_1 \cup \{\tau_1\}$ are uncontrollable, for any supervisory policy $\mathcal{P}$, $\exists \hat{\mathbf{m}} \in \Re(\tilde{M}, \tilde{\mathbf{m}}^0, \mathcal{P})$ such that:

- $\forall p \in \Pi_1, \hat{\mathbf{m}}(p) = \mathbf{m}(p)$;
- $\forall p \in \Pi_2, \hat{\mathbf{m}}(p) = \mathbf{m}_2^0(p)$;
- $\hat{\mathbf{m}}(\pi_2) = 1$;
- $\forall p \in \tilde{\Pi} - \{\Pi_1 \cup \Pi_2 \cup \{\pi_2\}\}, \hat{\mathbf{m}}(p) = 0$.

Since $\mathbf{m} \notin \Re(N_2, \mathbf{m}_2^0)$, for any marking $\tilde{\mathbf{m}}$ following the first firing of τ_2, $\exists\, i \in \{1, 2, \cdots, n\}$ such that $\tilde{\mathbf{m}}(p_i^1) \neq \tilde{\mathbf{m}}(p_i^2)$. This, in turn, implies there does not exist a supervisory policy that enforces liveness in $\tilde{M}$.

(If Part): Let $\Re(N_1, \mathbf{m}_1^0) \subseteq \Re(N_2, \mathbf{m}_2^0)$. The first firing of τ_1 will remove the token in π_1 and place it in π_2. The absence of a token in π_1 will (temporarily) freeze the token-load of those places of $\tilde{M}$ that originally belonged to Π_1. Since $\Re(N_1, \mathbf{m}_1^0) \subseteq \Re(N_2, \mathbf{m}_2^0)$, $\exists$ as valid firing sequence $\sigma \in T_2^*$ whose firing results in the token-load of p_i^2 equal the token-load of p_i^1, $\forall\, i \in \{1, 2, \cdots, n\}$. Since all transitions in $T_2 \cup \{\tau_2\}$ are controllable, by the appropriate choice of enabled transitions for each marking, we can guarantee the validity of only σ and no other firing sequence, under supervision. Immediately after this marking is reached, where the token load of a place originally in Π_1 is identical to the token-load of the corresponding place originally in Π_2, we control-disable all transitions in T_2 and control-enable τ_2. This way we guarantee the fact that the first arrival of a token in π_3 is accompanied by the fact that the token-load of a place originally in Π_1 is identical to the token-load of the corresponding place in Π_2. When the token arrives for the first time to $\pi_{i+2} (i \in \{1, 2, \cdots, n\})$, the controllable transition τ_{i+2} is control-disabled until the token-load of p_i^1 and p_i^2 equals zero. This is guaranteed as these places have an identical token-load and the repeated (uncontrollable) firing of $\hat{\tau}_i^1 \hat{\tau}_i^2$ can guarantee the reachability of the above submarking. Following the emptying of p_i^1 and p_i^2, the controllable transition τ_{i+2} can be enabled, at which point the token in π_{i+2} can move to π_{i+3}. Since all places in $\{p_i^j | j \in \{1, 2\}, i \in \{1, 2, \cdots, n\}\}$ are empty when the token arrives for the first time in π_{n+3}, the policy enforces liveness in $\hat{M}$ □

Theorem 5.3 leads to the following corollary.

Corollary 5.2: The inclusion problem for PN reachable sets reduces to the existence of a supervisory policy that enforces liveness in an arbitrary CtlPN. Therefore, the existence of a supervisory policy that enforces liveness in an arbitrary CtlPN is undecidable (cf. [14, Th. 5.11] and [17, Th. 6.3]).

We now consider the problem of deciding the existence of a supervisory control policy that enforces liveness in bounded CtlPN's. In Corollary 5.3 we present a necessary and sufficient condition for the existence of a supervisory policy that enforces liveness. Before we present the detailed proof, we first present a procedure *test_condition* (M) in Fig. 9 that uses the coverability graph $\hat{G}(N, \mathbf{m}^0) = (\hat{V}, \hat{A}, \hat{\Psi})$ of the underlying PN $N = (\Pi, T_u \cup T_c, \Phi, \mathbf{m}^0)$ of a CtlPN $M = (\Pi, T_u, T_c, \Phi, \mathbf{m}^0, C, B)$. For each vertex $v_i \in \tilde{V}$, where $\tilde{V} \subseteq \hat{V}$, let $\Delta(v_i, \tilde{V}) = \{v_j \in \tilde{V} | \exists\, \sigma_1 \sigma_2 \in T^*$ such that:

1) $v_i \rightarrow \sigma_1 \rightarrow v_j \rightarrow \sigma_2 \rightarrow v_j$;
2) $\mathbf{x}(\sigma_2) > 0$;
3) $\mathbf{C}\mathbf{x}(\sigma_2) \geq 0$;
4) $\forall \sigma_3 \in pr(\sigma_1 \sigma_2)$

$(v_i \rightarrow \sigma_3 \rightarrow v_k \Rightarrow v_k \in \tilde{V})\}$

where $pr(\bullet)$ denotes the prefix-set of the string argument. From this definition it follows that $\forall\, v_i \in \tilde{V}$

$$\Delta(v_i, \tilde{V}) = \bigcup_{v_j \in (v_i^\bullet)^\bullet - \{v_i\}} \Delta(v_j, \tilde{V}). \tag{4}$$

We now present some observations on procedure *test_condition* (M).

Observation 5.1: The procedure *test_condition* (M) terminates in finite time.

Proof: If the while-loops in the above procedure are never executed, proper termination is trivially guaranteed. The proper termination of the first while-loop can be inferred from the fact that the coverability graph has a finite number of vertices and the fact that the procedure does not convert the label of a vertex from "illegal" to "legal." To complete this observation, we observe that at the end of each execution of the first while-loop the size of the set $\mathcal{A} = \{a \in \hat{A} | \Gamma(a) \in T_u$ and ${}^\bullet a$ is labeled "legal" and $a^\bullet$ is labeled "illegal."$\}$ is reduced by unity. For the finite coverability graph with a finite number of arcs, this will eventually result in $\mathcal{A}$ being empty.

During each execution of the second while-loop, either the number of vertices labeled "legal" that are connected to v_0 at the end of each execution will be fewer than the corresponding number at the start of execution, or the number of edges

```
test_condition(M)
CtlPN M = (Π, T_u, T_c, Φ, m^0, C, B);
{
    Construct the coverability graph Ĝ(N, m^0) = (V̂, Â, Ψ̂) of the underlying PN N = (Π, T_u ∪ T_c,
    Φ, m^0) of M;
    /* Let Γ: Â → T (where T = T_u ∪ T_c), be the function that assigns transitions to each edge in
    the coverability graph. */
    For each v_i ∈ V̂ do {
        /* use the procedure outlined in the appendix */
        if (Δ(v_i, V̂) ≠ ∅)
            v_i is labeled "legal;"
        else
            v_i is labeled "illegal;"
    }
    while ( (∃ a ∈ Â, such that Γ(a) ∈ T_u) && (Ψ̂(a) = (v_i, v_j), where v_i labeled "legal" and v_j is
    labeled "illegal") )
        the label of v_i is converted from "legal" to "illegal;"
    /* Following this step all edges that originate from a "legal" vertex and terminate at an "illegal"
    vertex have controllable transitions associated with them */
    while ( (v_0 is labeled "legal") && (∃ v_l that is labeled "illegal" that is connected to v_0) ) {
        For each a ∈ Â do {
            if ( (Γ(a) ∈ T_c) && (Ψ̂(a) = (v_i, v_j), where v_i is labeled "legal" and v_j is
            labeled "illegal") )
                if (∃ v_k ∈ V̂, and ∃ â ∈ Â such that Ψ̂(â) = (v_i, v_k) and v_k is
                labeled "legal")
                    delete the edge a;
                    modify Â and Ψ̂ accordingly;
                else
                    the label of v_i is converted from "legal" to "illegal;"
        }
    }
    if (v_0 is labeled "legal")
        return TRUE;
    else
        return FALSE;
}
```

Fig. 9. The procedure *test_condition* (M), where M is a CtlPN.

connected to v_0 at the end of an execution will be fewer than the corresponding number at the start of the execution. Therefore, this process will culminate in all vertices connected to v_0 being labeled "legal," or v_0 being labeled "illegal," at which point the second while-loop will be exited, guaranteeing proper termination. □

Observation 5.2: At any point in the execution of test_condition (M), if a vertex v_i is connected to v_0 and is labeled "legal," then $\exists v_j \in \Delta(v_i, \hat{V}_{con})$, where $\hat{V}_{con} \subset \hat{V}$ is the set of vertices that are currently connected to v_0.

Proof: If a vertex v_i is assigned a "legal" label before the execution of the while-loops, the statement of the observation is true. Since no edges are removed in the first while-loop, the statement of the observation holds at the termination of the first while-loop also.

As the induction hypothesis we assume that at the beginning of each execution of any statement in the second while-loop, the statement of the observation holds. The base case was established before entry into the second while-loop. If no edges are deleted at the current statement, the statement of the observation holds at the conclusion of the current statement also. If an edge a is deleted in the current statement, then the following conditions must be true: $\exists a \in \hat{A}$, such that $\Gamma(a) \in T_c$; ${}^\bullet a (= v_k$, say) is labeled "legal;" $a^\bullet (= v_l$, say) is labeled "illegal;" and $\exists \hat{a} \in \hat{A}$, such that ${}^\bullet \hat{a} = v_k$ and $\hat{a}^\bullet (= v_m$, say) is labeled "legal." Let $\hat{V}_{con}^{pre} (\hat{V}_{con}^{post})$ be the set

of vertices connected to v_0 before (after) the execution of the current statement.

By the induction hypothesis $\exists v_n \in \Delta(v_m, \hat{V}_{con}^{pre})$ that is connected to v_m. From the definition it follows that $v_n \in \Delta(v_k, \hat{V}_{con}^{pre})$ [cf. (4)]. Since the deleted edge a originates from v_k and terminates at $v_l (\neq v_m)$, it follows that $v_n \in \Delta(v_k, \hat{V}_{con}^{post})$.

Note that $v_i \in \hat{V}_{con}^{post} \Rightarrow v_i \in \hat{V}_{con}^{pre}$. We consider two cases in the subgraph induced by $\hat{V}_{con}^{pre}$: 1) $\exists\ \sigma \in \hat{A}^*$, such that $v_i \rightarrow \sigma \rightarrow v_k$, and 2) $\nexists \sigma \in \hat{A}^*$, such that $v_i \rightarrow \sigma \rightarrow v_k$.

For case 1), without loss in generality we can assume the path identified by σ does not involve traversals of the edge a that is to be deleted. So, upon the deletion of the edge a, the path identified by σ would remain intact, and from the definition we infer $v_n \in \Delta(v_i, \hat{V}_{con}^{post})$. For case 2), we note that $\forall v_p \in \Delta(v_i, \hat{V}_{con}^{pre}), \forall \sigma_1\sigma_2 \in \hat{A}^*$, such that $v_i \rightarrow \sigma_1 \rightarrow v_p \rightarrow \sigma_2 \rightarrow v_p$, the deleted edge a does not appear anywhere in $\sigma_1\sigma_2$. This is because there are no paths from v_i to v_k. We infer that $\Delta(v_i, \hat{V}_{con}^{pre}) = \Delta(v_i, \hat{V}_{con}^{post})$. Thus establishing the induction step for $v_i \in \hat{V}_{con}^{post}$. Hence the observation. □

The following observation is stated without proof and results directly from the exit condition of the second while-loop.

Observation 5.3: If *test_condition* (M) returns TRUE, then at the termination of the procedure all vertices connected to v_0 are labeled "legal."

Theorem 5.4: Let $\hat{G}(N, \mathbf{m}^0) = (\hat{V}, \hat{A}, \hat{\Psi})$ be the coverability graph of the underlying PN $N = (\Pi, T_u \cup T_c, \Phi, \mathbf{m}^0)$ of a CtlPN $M = (\Pi, T_u, T_c, \Phi, \mathbf{m}^0, C, B)$. $\exists$ a control invariant subset of vertices $\mathcal{V} \subseteq \hat{V}$ such that:

1) $v_0 \in \mathcal{V}$;
2) $\forall v \in \mathcal{V}, \exists \hat{v} \in \mathcal{V}, \exists \sigma_1\sigma_2 \in T^*$:
   a) $v \rightarrow \sigma_1 \rightarrow \hat{v} \rightarrow \sigma_2 \rightarrow \hat{v}, \mathbf{x}(\sigma_2) > 0$, and $\mathbf{Cx}(\sigma_2) \geq 0$;
   b) $\forall \sigma_3 \in pr(\sigma_1\sigma_2), v \rightarrow \sigma_3 \rightarrow \tilde{v} \Rightarrow \tilde{v} \in \mathcal{V}$, where $pr(\bullet)$ denotes the prefix-set of the string argument, if and only if *test_condition* (M) returns TRUE.

Proof (Only If Part): Let $\mathcal{V} \subseteq \hat{V}$ be a control invariant subset of vertices such that the conditions in the statement of the theorem are true. We will show that *test_condition* (M) returns TRUE by an induction argument.

All members of $\mathcal{V}$, and possibly some members of $\hat{V} - \mathcal{V}$, will be assigned a "legal" label before the first while-loop is executed. Also, note that $\forall v_i \in \mathcal{V}, \exists v_j \in \mathcal{V}, \exists a \in \hat{A}$, such that $\hat{\Psi}(a) = (v_i, v_j)$.

Since $\mathcal{V}$ is control invariant, if $v_i \in \mathcal{V}, \exists t_u \in T_u$ such that $v_i \rightarrow t_u \rightarrow v_j \Rightarrow v_j \in \mathcal{V}$. Therefore, after the termination of the first while-loop the labels of the members of $\mathcal{V}$ will remain unaltered. Additionally, $\forall \hat{v} \in \mathcal{V}, \exists \tilde{v} \in \mathcal{V}, \exists a \in \hat{A}$, such that $\hat{\Psi}(a) = (\hat{v}, \tilde{v})$. This serves as the base case for the induction argument needed for the second while-loop.

As in the induction hypothesis, assume every element of $\mathcal{V}$ is assigned a label of "legal" and $\forall \hat{v} \in \mathcal{V}, \exists \tilde{v} \in \mathcal{V}, \exists a \in \hat{A}$ such that $\hat{\Psi}(a) = (\hat{v}, \tilde{v})$, before the execution of some statement in the second while-loop. Let $v_i \in \mathcal{V}$ be assigned the label "legal," while $v_j \in \hat{V}$ is assigned the label "illegal." Also, let $\exists a \in A$ such that $\hat{\Psi}(a) = (v_i, v_j)$. By the induction hypothesis, $\exists v_k \in \mathcal{V}$, such that v_k is assigned the label "legal." This will result in the edge a being deleted, and since this edge originates from a vertex labeled "legal" to a vertex labeled "illegal," its deletion will not influence the property that $\forall \hat{v} \in \mathcal{V}, \exists \tilde{v} \in \mathcal{V}, \exists a \in \hat{A}$, such that $\hat{\Psi}(a) = (\hat{v}, \tilde{v})$, thus establishing the induction step.

From Observation 5.1 we know the procedure terminates in finite time. Since $v_0 \in \mathcal{V}$, and the labels of vertices in $\mathcal{V}$ are not altered, it follows that upon termination, the label assigned to v_0 remains "legal." Therefore, *test_condition* (M) returns TRUE.

(If Part): From Observation 5.3 we know that if *test_condition* (M) returns TRUE, all vertices connected to v_0 at the termination of *test_condition* (M), denoted by $\mathcal{V}$, are labeled "legal." Clearly, $v_0 \in \mathcal{V}$. From Observation 5.2 we know that $\forall v \in \mathcal{V}, \exists \hat{v} \in \mathcal{V}, \exists \sigma_1\sigma_2 \in T^*, v \rightarrow \sigma_1 \rightarrow \hat{v} \rightarrow \sigma_2 \rightarrow \hat{v}, \mathbf{x}(\sigma_2) > 0, \mathbf{Cx}(\sigma_2) \geq 0$, and $\forall \sigma_3 \in pr(\sigma_1\sigma_2), (v \rightarrow \sigma_3 \rightarrow \tilde{v} \Rightarrow \tilde{v} \in \mathcal{V})$. The control invariance of $\mathcal{V}$ follows from the fact that if $\exists v \in \mathcal{V}, \exists a \in \hat{A}, \exists \hat{v} \in \hat{V} - \mathcal{V}$, such that $v \rightarrow \Gamma(a) \rightarrow \hat{v}$ in $\hat{G}(N, \mathbf{m}^0)$, then $\Gamma(a) \in T_c$ and hence the result. □

Corollary 5.3 follows from Theorems 5.1 and 5.4 and the fact that for bounded PN's there is a one-to-one correspondence between the vertices in the coverability graph and the reachable marking set.

Corollary 5.3: There exists a supervisory policy that enforces liveness in a bounded CtlPN $M = (\Pi, T_u, T_c, \Phi, \mathbf{m}^0, C, B)$ if and only if the procedure *test_condition* (M) returns TRUE.

If $M = (\Pi, T_u, T_c, \Phi, \mathbf{m}^0, C, B)$ is a bounded CtlPN, or a CtlPN with an empty set of uncontrollable transitions, such that *test_condition* (M) returns TRUE, the following supervisory policy enforces liveness and is minimally restrictive: permit the firing of $t_c \in T_c$ at the marking $\hat{\mathbf{m}}$ if and only if *test_condition* $(\tilde{M})$ returns TRUE, where $\tilde{M} = (\Pi, T_u, T_c, \Phi, \tilde{\mathbf{m}}, C, B)$ and $\hat{\mathbf{m}} \rightarrow t_c \rightarrow \tilde{\mathbf{m}}$. This is described concisely in Fig. 10. This supervisory policy is minimally restrictive. To see this, note that at any given marking, a transition is disabled only when its firing would result in a marking from which it would not be possible to fire every transition indefinitely often, which is a necessary condition for liveness.

VI. Discussion

The conditions of Theorem 5.1 imply that if there exists a supervisory policy that enforces liveness in an arbitrary CtlPN M, then the underlying PN N has to be *repeatable*. A PN N is said to be *repeatable* if and only if $\exists \mathbf{x} > 0$ (i.e., each component of $\mathbf{x}$ is strictly greater than zero) such that $\mathbf{Cx} \geq 0$ (cf. [13, Sec. 8]). This is a direct consequence of Conditions 1a) and b) of Theorem 5.1. However, repeatability is not sufficient for a supervisory policy to enforce liveness. As we have seen in Section III, the initial marking and the set of uncontrollable transitions play a critical role in deciding the existence of a supervisory policy that enforces liveness.

Unlike our definition of a PN or CtlPN that includes the initial marking, some researchers in PN theory do not include

```
compute_control (m)
marking m;
{
    For each t_i ∈ T_u do:
        u_i = 1;
    For each t_j ∈ T_c do:
        if ( (t_j ∉ T_e(m)) || (test_condtion (M̃) returns TRUE, where M̃ = (Π, T_u, T_c, Φ, m̃,
        C, B), and m → t_c → m̃) )
            u_i = 1;
        else
            u_i = 0;
}
```

Fig. 10. A minimally restrictive supervisory policy that enforces liveness for a bounded CtlPN, or an unbounded CtlPN, where $T_u = \emptyset$, $M = (\Pi, T_u, T_c, \Phi, \mathbf{m}^0, C, B)$ such that *test_condition* (M) returns TRUE.

the initial marking as a part of the definition. The initial marking can be chosen at initialization. For this special case the repeatability of the underlying PN of the CtlPN is also sufficient for the existence of a supervisory control policy that enforces liveness, provided the CtlPN has no uncontrollable transitions. To see this, observe that for a CtlPN M with a repeatable underlying PN N, there is an m-dimensional, integer-valued vector, $\mathbf{x} > 0$, with strictly positive components such that $\mathbf{Cx} \geq 0$. We can choose an initial marking $\mathbf{m}^0$ such that there is a valid firing sequence σ, starting from $\mathbf{m}^0$, such that the Parikh mapping of σ is the vector $\mathbf{x}$. One choice of $\mathbf{m}^0$ could be

$$\mathbf{m}^0(p) = \sum_{t_i \in p^\bullet} \mathbf{x}_i \tag{5}$$

where $\mathbf{x}_i$ is the ith component of the vector $\mathbf{x}$. Since each transition in the CtlPN can be individually controlled, we can restrict the firing of transitions to enforce a firing vector of $\mathbf{x}$ using an appropriately defined supervisory policy. One such policy is the round-robin policy where in each cycle, transition t_i is allowed to fire no more than $\mathbf{x}_i$ times. This supervisory policy enforces liveness but is not necessarily minimally restrictive.

From Farkas' lemma it follows that if for a PN N, $\exists y \geq 0$, such that $y^T C \leq 0$, and some element of $\mathbf{y}^T\mathbf{C}$ is strictly less than zero, then N is nonrepeatable. The underlying PN of the CtlPN shown in Fig. 4(a) is not repeatable as $y^T C = (-1\ 0\ 0)^T$, for $y = (1\ 1\ 1)^T$, and the underlying PN of the CtlPN shown in Fig. 4(b) can be shown to be nonrepeatable for $y = (1)$. Similarly, the underlying PN of the CtlPN shown in Fig. 5 is not repeatable as $y^T C = (0\,0 -1\,0\,0\,0 -1\,0\,0\,0)^T$, for $y = (2\ 2\ 1\ 1\ 1\ 1\ 1\ 1\ 2)^T$. Therefore, for these CtlPN's there are no supervisory control policies that can enforce liveness.

We now consider the CtlPN shown in Fig. 6. The set of uncontrollable transitions for this unbounded CtlPN is empty. The KM-tree and coverability graph where the root vertex is assigned the marking $\mathbf{m^0} = (1\ 0\ 0\ 0)^T$ is shown in Fig. 7. The procedure *test_condition* $(\mathbf{m^0})$ listed in Fig. 9 returns TRUE, as all vertices in the coverability graph shown in Fig. 7 are labeled "legal." To see this observe that there is a path $\sigma = t_1 t_1 t_2 t_3 t_4$ from v_1 to v_1 that involves all transitions and v_1 is connected to v_0. Also, $\mathbf{Cx}(\sigma) = (0\ 0\ 0\ 1)^T \geq 0$. By Theorem 5.2 and Corollary 5.1 we infer the existence of a supervisory policy that enforces liveness. Using the results of Theorem 5.2 and Corollary 5.1 we show that the supervisory policy described in Fig. 10 when applied to the CtlPN shown in Fig. 6 will not allow the firing of the sequence $t_1 t_2 t_3$. Transition t_1 is state-enabled at the marking $\mathbf{m^0} = (1\,0\,0\,0)^T$. Its firing will result in the marking $\mathbf{m^1} = (1\ 0\ 1\ 1)^T$. To investigate the control-enabling of t_1 under the marking $\mathbf{m^0}$, we need the KM-tree and coverability graph of the underlying PN with an initial marking $\mathbf{m^1} = (1\ 0\ 1\ 1)^T$. These are shown in Fig. 11. We now investigate the execution of procedure *test_condition* $(\mathbf{m^1})$. We observe that $v_1 \rightarrow t_1 t_1 t_2 t_3 t_4 \rightarrow v_1$. Also, $\mathbf{Cx}(t_1 t_1 t_2 t_3 t_4) = (0\ 0\ 0\ 1)^T$. Therefore, the vertex labeling before the execution of the while-loops is as follows: vertices v_0, v_1, v_3, v_6 and v_8 are labeled "legal," while vertices v_7 and v_{10} are labeled "illegal." The second while-loop is executed once, which results in the deletion of the edges $\hat{a}$ and $\tilde{a}$, where $\hat{\Psi}(\hat{a}) = (v_6, v_7)$ and $\hat{\Psi}(\tilde{a}) = (v_8, v_{10})$. Following the deletion of these edges, all vertices connected to v_0 will be labeled "legal," and the second while-loop is exited. Since v_0 is labeled "legal," procedure *test_condition* $(\mathbf{m^1})$ returns TRUE. So, t_1 is control-enabled under the marking $\mathbf{m^0}$. Since it is the only transition that is state-enabled under $\mathbf{m^0}$, it will be the only transition that can fire at some subsequent instance in time. Its firing will result in the marking $\mathbf{m^1} = (1\ 0\ 1\ 1)^T$, and $T_e(\mathbf{m^1}) = \{t_1, t_2\}$. Since we are investigating the firability under supervision of the string $t_1 t_2 t_3$, we restrict our attention to investigating the control-enabling of t_2 only. The marking that will result from the firing of t_2 under $\mathbf{m^1}$ is $\mathbf{m^2} = (0\ 1\ 1\ 1)^T$. The KM-tree and coverability graph with an initial marking of $\mathbf{m^2}$ is shown in Fig. 12. The procedure *test_condition* $(\mathbf{m^2})$ returns TRUE. To see this, observe that prior to the execution of the first while-loop vertices v_0, v_2, v_3, and v_5 are labeled

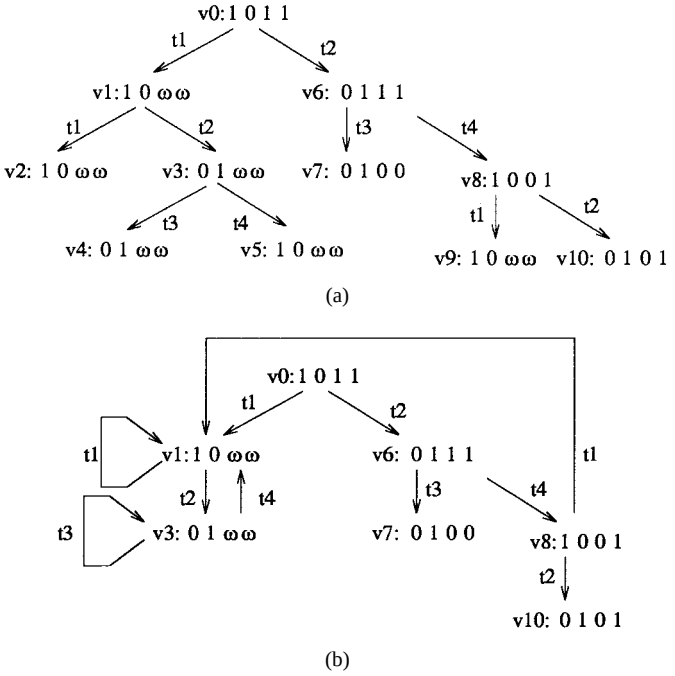

Fig. 11. (a) The KM-tree and (b) coverability graph for a PN with the same structure as the PN shown in Fig. 6, with initial marking $(1\ 0\ 1\ 1)^T$.

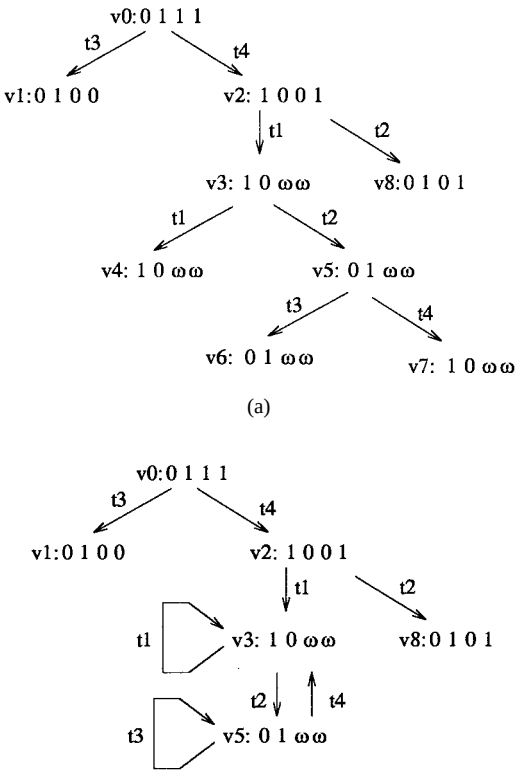

Fig. 12. (a) The KM-tree and (b) coverability graph for a PN with the same structure as the PN shown in Fig. 6 with initial marking $(0\ 1\ 1\ 1)^T$.

"legal," while vertices v_1 and v_8 are labeled illegal. The execution of the second while-loop results in the deletion of edges $\hat{a}$ and $\tilde{a}$, where $\hat{\Psi}(\hat{a}) = (v_0, v_1)$ and $\hat{\Psi}(\tilde{a}) = (v_2, v_8)$. Following the deletion of these edges, all vertices connected to v_0 are labeled "legal" and *test_condition* $(\mathbf{m^2})$ returns TRUE. So, t_2 is control-enabled and state-enabled under the marking $\mathbf{m^1}$, and its firing at some subsequent instant in time will yield the marking $\mathbf{m^2}$, and $T_e(\mathbf{m^2}) = \{t_3, t_4\}$. If t_3 is permitted to fire, the resulting marking would be $\mathbf{m^3} = (0\ 1\ 0\ 0)^T$, and $T_e(\mathbf{m^3}) = \emptyset$. Therefore, the KM-tree and coverability graph with an initial marking of $\mathbf{m^3}$ will contain a singleton vertex and no arcs. Clearly, *test_condition* $(\mathbf{m^3})$ returns FALSE, so t_3 is not control-enabled under the marking $\mathbf{m^2}$. Therefore, the sequence $t_1t_2t_3$ is not valid under supervision. It can be shown that the transition t_4 is control-enabled under the marking $\mathbf{m^2}$. Proceeding this way *ad infinitum*, it can be shown that the supervisory policy in Fig. 10 enforces liveness.

If the problem of enforcing liveness is interpreted as a forbidden-state (forbidden-marking) problem, Theorems 5.1 and 5.2 and Corollary 5.1 provide the definition of the set of marked, legal-states $\mathcal{M}_{legal}$, where

$$\mathcal{M}_{legal} := \{\mathbf{m} \in \Re(N, \mathbf{m}^0) | \mathit{test\_condition}\,(\mathbf{m}) \text{ returns TRUE}\}$$

provided the CtlPN has an empty set of uncontrollable transitions. For this case, it can be shown that $\mathcal{M}_{legal} \neq \emptyset \Leftrightarrow$ *test_condition* $(\mathbf{m}^0)$ returns TRUE. Also, it can be shown that if $\mathcal{M}_{legal} \neq \emptyset$, then every marking in $\mathcal{M}_{legal}$ is reachable from $\mathbf{m}^0$. Controllability of $\mathcal{M}_{legal}$ is equivalent to the control invariance of $\mathcal{M}_{legal}$. For the class of CtlPN's under consideration, the control invariance of $\mathcal{M}_{legal}$ follows trivially from the fact that in a CtlPN every transition is individually controllable. In Section IV we had mentioned that there is a conceptual difference between the forbidden-marking specifications and specifications for liveness. We now qualify this statement for CtlPN's where not all transitions are controllable. As an illustration we consider a CtlPN with an underlying PN as shown in Fig. 1. In the absence of supervision, the set of reachable markings of the underlying PN is $\{(1\ 0\ 0\ 0\ 1)^T, (0\ 1\ 0\ 1\ 0)^T, (1\ 0\ 1\ 0\ 0)^T, (1\ 0\ 0\ 1\ 0)^T, (0\ 1\ 0\ 0\ 1)^T\}$, the set of legal-states $\mathcal{M}_{legal} = \{(10001)^T, (0\ 1\ 0\ 1\ 0)^T, (1\ 0\ 1\ 0\ 0)^T, (1\ 0\ 0\ 1\ 0)^T\}$. Let t_4 be the only uncontrollable transition. It is not hard to see that $\mathcal{M}_{legal}$ is not control invariant. The supremal controllable subset of $\mathcal{M}_{legal}$ is the singleton $\{(1\ 0\ 0\ 0\ 1)^T\}$. Any supervisory policy that restricts the reachable states under supervision to a (supremal) controllable subset of $\mathcal{M}_{legal}$ will not achieve the objective of enforcing liveness. If t_4 is not controllable, there does not exist any supervisory policy that can enforce liveness. This is a critical divergence from conventional forbidden state problems, where it is assumed that the desired specification always supports the partial order induced by containment on the set of reachable states, thus implying the desirability of the supremal controllable subset.

If $\mathcal{P}_1$ and $\mathcal{P}_2$ are two supervisory policies that enforce liveness and if neither policy is less restrictive than the other, then the supervisory policy $\tilde{\mathcal{P}}$: $\mathcal{N}^n \rightarrow \{0, 1\}^m$, de-

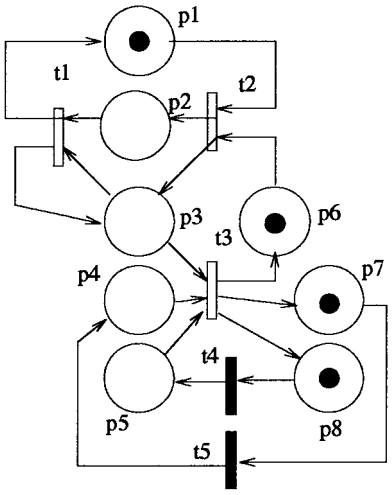

Fig. 13. An illustration of Theorem 6.1 using the supervisory policies shown in Fig. 14.

fined as shown in (6), at the bottom of the page, where $i \in \{1, 2, \cdots, m\}$, will be less restrictive than $\mathcal{P}_1$ and $\mathcal{P}_2$. In Theorem 6.1 we show that $\tilde{\mathcal{P}}$ also enforces liveness in M. This implies that a minimally restrictive supervisory policy that enforces liveness exists if and only if there is a supervisory policy that enforces liveness in a CtlPN.

Theorem 6.1: If $\mathcal{P}_i$: $\mathcal{N}^n \rightarrow 0, 1^m (i = 1, 2)$ are two different supervisory policies that enforce liveness in a CtlPN $M = (\Pi, T_u, T_c, \Phi, \mathbf{m}^0, C, B)$, then the supervisory policy $\tilde{\mathcal{P}}$ defined above also enforces liveness in the CtlPN M.

Proof: We first show that $\Re(M, \mathbf{m}^0, \tilde{\mathcal{P}}) = \Re(M, \mathbf{m}^0, \mathcal{P}_1) \cup \Re(M, \mathbf{m}^0, \mathcal{P}_2)$. The fact that $\Re(M, \mathbf{m}^0, \tilde{\mathcal{P}}) \subseteq \Re(M, \mathbf{m}^0, \mathcal{P}_1) \cup \Re(M, \mathbf{m}^0, \mathcal{P}_2)$ is established by induction over the length of firing sequences valid under the supervision of $\tilde{\mathcal{P}}$.

As the base case we observe that $\mathbf{m}^0 \in \Re(M, \mathbf{m}^0, \tilde{\mathcal{P}})$ and $\mathbf{m}^0 \in \Re(M, \mathbf{m}^0, \mathcal{P}_1) \cup \Re(M, \mathbf{m}^0, \mathcal{P}_2)$. As in the induction hypothesis, we note that if $\mathbf{m}^0 \rightarrow \sigma \rightarrow \mathbf{m}$ under the supervision of $\tilde{\mathcal{P}}$, then $\mathbf{m} \in \Re(M, \mathbf{m}, \mathcal{P}_1) \cup \Re(M, \mathbf{m}^0, \mathcal{P}_2)$. For the induction step, let $\mathbf{m} \rightarrow t_i \rightarrow \hat{\mathbf{m}}$ under the supervision of $\tilde{\mathcal{P}}$. By the induction hypotheses we infer either $\mathbf{m} \in \Re(M, \mathbf{m}^0, \mathcal{P}_1) \cap \Re(M, \mathbf{m}^0, \mathcal{P}_2)$, or $\mathbf{m} \in \Re(M, \mathbf{m}^0, \mathcal{P}_1) - \Re(M, \mathbf{m}^0, \mathcal{P}_2)$, or $\mathbf{m} \in \Re(M, \mathbf{m}^0, \mathcal{P}_2) - \Re(M, \mathbf{m}^0, \mathcal{P}_1)$. In each of these cases, using the definition of $\tilde{\mathcal{P}}$ it can be shown that $\hat{\mathbf{m}} \in \Re(M, \mathbf{m}, \mathcal{P}_1) \cup \Re(M, \mathbf{m}^0, \mathcal{P}_2)$. Therefore, $\Re(M, \mathbf{m}^0, \tilde{\mathcal{P}}) \subseteq \Re(M, \mathbf{m}^0, \mathcal{P}_1) \cup \Re(M, \mathbf{m}^0, \mathcal{P}_2)$.

The fact that $\Re(M, \mathbf{m}^0, \mathcal{P}_1) \cup \Re(M, \mathbf{m}^0, \mathcal{P}_2) \subseteq \Re(M, \mathbf{m}^0, \tilde{\mathcal{P}})$ follows directly from the definition of $\tilde{\mathcal{P}}$, and the routine details of the proof are skipped for brevity.

Since $\mathcal{P}_1$ and $\mathcal{P}_2$ enforce liveness, it follows that $\forall i \in \{1, 2\}, \forall t_j \in T, \forall \mathbf{m} \in \Re(M, \mathbf{m}^0, \mathcal{P}_i), \exists \hat{\mathbf{m}} \in \Re(M, \mathbf{m}^0, \mathcal{P}_i)$ such that $t_j \in T_e(\hat{\mathbf{m}})$ and $\mathcal{P}_i(\mathbf{m})_j = 1$. This observation with the fact that $\Re(M, \mathbf{m}^0, \tilde{\mathcal{P}}) = \Re(M, \mathbf{m}^0, \mathcal{P}_1) \cup \Re(M, \mathbf{m}^0, \mathcal{P}_2)$ is sufficient to establish the fact that $\tilde{\mathcal{P}}$ enforces liveness in M, hence the result. □

As an illustration of Theorem 6.1, we consider the CtlPN shown in Fig. 13. Transitions t_4 and t_5 are controllable transitions, and transitions t_1, t_2, and t_3 are uncontrollable. The sixteen states of the underlying PN of this CtlPN are listed in the table shown in Fig. 14. This CtlPN can be made live by guaranteeing the nonreachability of the markings $\{(0\ 1\ 1\ 1\ 1\ 0\ 0\ 0)^T, (1\ 0\ 0\ 1\ 1\ 1\ 0\ 0)^T\}$ under supervision. Supervisory policy $\mathcal{P}_1(\mathcal{P}_2)$ accomplishes this objective by preventing the firing of the controllable transition $t_5(t_4)$ when place p_2 or p_6 has a token. The supervisory policy $\tilde{\mathcal{P}}$ obtained from $\mathcal{P}_1$ and $\mathcal{P}_2$ using (6) is enumerated in the last column of the table in Fig. 14. The supervisory policy $\tilde{\mathcal{P}}$ enforces liveness in the CtlPN shown in Fig. 13, and it is less restrictive than $\mathcal{P}_1$ and $\mathcal{P}_2$.

We turn our attention to the complexity of deciding the existence of supervisory policies that enforce liveness in an arbitrary CtlPN. First, we emphasize that in this paper we are concerned primarily with the solvability or unsolvability of the above problem under various restrictions. By Corollary 5.2 we infer the problem is generally unsolvable. If the CtlPN is bounded, by Corollary 5.3 we infer the existence of a supervisor that enforces liveness can be decided in finite time. The outlined approach in Section V and the Appendix requires the construction of a coverability graph of the underlying PN followed by additional computations. The number of vertices in a coverability graph of a PN can be exponentially related to the number of places and transitions. Additionally, the procedure outlined in the Appendix involves the solution of a linear program. Although the time complexity of a linear program is polynomially related to the number of equations and variables [21], the number of equations and variables in our case can be exponentially related to the number of places and transitions. Consequently, the computational complexity of the procedures that investigate the existence of a supervisory policy that enforces liveness is discouraging. This should come as no surprise as the test for the existence of a supervisor that enforces liveness in an arbitrary CtlPN is at least as difficult as the test for liveness of its underlying PN. To establish this reduction, we note that any arbitrary PN can be interpreted as a CtlPN where are all transitions are uncontrollable. There exists a supervisory policy (i.e., the trivial supervisory policy) that enforces liveness in this CtlPN if and only if the original

$$\tilde{\mathcal{P}}(\mathbf{m})_i = \begin{cases} \mathcal{P}_1(\mathbf{m})_1 \vee \mathcal{P}_2(\mathbf{m})_1, & \text{if } \mathbf{m} \in \Re(M, \mathbf{m}^0, \mathcal{P}_1) \cap \Re(M, \mathbf{m}^0, \mathcal{P}_2), \\ \mathcal{P}_1(\mathbf{m})_i, & \text{if } \mathbf{m} \in \Re(M, \mathbf{m}^0, \mathcal{P}_1) - \Re(M, \mathbf{m}^0, \mathcal{P}_2), \\ \mathcal{P}_2(\mathbf{m})_i, & \text{if } \mathbf{m} \in \Re(M, \mathbf{m}^0, \mathcal{P}_2) - \Re(M, \mathbf{m}^0, \mathcal{P}_1), \\ \{1\}^m, & \text{otherwise} \end{cases} \tag{6}$$

marking $\mathbf{m}$	$\mathbf{m} \in \Re(M, \mathbf{m}^0, \mathcal{P}_1)$?	$\mathbf{m} \in \Re(M, \mathbf{m}^0, \mathcal{P}_2)$?	$\mathcal{P}_1(\mathbf{m})$	$\mathcal{P}_2(\mathbf{m})$	$\mathbf{m} \in \Re(M, \mathbf{m}^0, \tilde{\mathcal{P}})$?	$\tilde{\mathcal{P}}(\mathbf{m})$
$(10000111)^T$	Yes	Yes	$(11110)^T$	$(11101)^T$	Yes	$(11111)^T$
$(01100011)^T$	Yes	Yes	$(11110)^T$	$(11101)^T$	Yes	$(11111)^T$
$(10100011)^T$	Yes	Yes	$(11111)^T$	$(11111)^T$	Yes	$(11111)^T$
$(10101010)^T$	Yes	Yes	$(11111)^T$	$(11111)^T$	Yes	$(11111)^T$
$(10111000)^T$	Yes	Yes	$(11111)^T$	$(11111)^T$	Yes	$(11111)^T$
$(10110001)^T$	Yes	Yes	$(11111)^T$	$(11111)^T$	Yes	$(11111)^T$
$(01101010)^T$	Yes	No	$(11110)^T$	$(11111)^T$	Yes	$(11110)^T$
$(01111000)^T$	No	No	$(11111)^T$	$(11111)^T$	No	$(11111)^T$
$(01000111)^T$	No	No	$(11111)^T$	$(11111)^T$	No	$(11111)^T$
$(01001110)^T$	No	No	$(11111)^T$	$(11111)^T$	No	$(11111)^T$
$(01011100)^T$	No	No	$(11111)^T$	$(11111)^T$	No	$(11111)^T$
$(01001101)^T$	No	No	$(11111)^T$	$(11111)^T$	No	$(11111)^T$
$(01110001)^T$	No	Yes	$(11111)^T$	$(11101)^T$	Yes	$(11101)^T$
$(10001110)^T$	Yes	No	$(11110)^T$	$(11111)^T$	Yes	$(11110)^T$
$(10011100)^T$	No	No	$(11111)^T$	$(11111)^T$	No	$(11111)^T$
$(10010101)^T$	No	Yes	$(11111)^T$	$(11101)^T$	Yes	$(11101)^T$

Fig. 14. An illustration of Theorem 6.1 using the CtlPN shown in Fig. 13.

PN is live. The test for liveness for even 1-conservative PN's[5] is $PSPACE$-complete[6] (cf. [10, Table I]). So, the test for the existence of a supervisory policy that enforces liveness in an arbitrary, bounded CtlPN is at least $PSPACE$-complete.

Once the existence of a supervisory policy that enforces liveness is established, the next step involves the computation of the required control for each reachable marking. For bounded CtlPN's this can be done concurrently with the test for the existence of a supervisor, resulting in an exhaustive table that prescribes the control for each marking reachable under supervision. The size of this table would depend on the size of the set of reachable markings under supervision. However, this might not be possible for unbounded CtlPN's with an empty set of uncontrollable transitions. For this case, the procedures outlined in Section V require the construction of a coverability graph $\hat{G}(N, \mathbf{m})$ for each previously unreached, marking $\mathbf{m}$. The stepwise refinement and abstraction of CtlPN's along the lines of the similar problem for PN's [19] can alleviate the computational burden in this case. Additionally, it may be possible to obtain the coverability graph $\hat{G}(N, \mathbf{m}^2)$ from the coverability graph $\hat{G}(N, \mathbf{m}^1)$ by minor modifications, where $\mathbf{m}^1 \rightarrow t_i \rightarrow \mathbf{m}^2$ for some transition t_i, thus avoiding the construction of the entire coverability graph for the marking $\mathbf{m}^2$. We suggest investigations into these approaches for future research.

VII. Conclusion

In this paper, we presented a necessary and sufficient condition for the existence of a supervisory policy that enforces liveness in arbitrary, CtlPN's. For CtlPN's where each transition can be individually controlled, we present a necessary and sufficient condition for the existence of a supervisor that enforces liveness. Also, as a part of the proof of sufficiency, we introduced a minimally restrictive supervisory policy that enforces liveness in this class of CtlPN's. For CtlPN's with a nonempty set of uncontrollable transitions, we showed the problem of deciding the existence of supervisory policies that enforce liveness is unsolvable. For bounded CtlPN's we derived a computable necessary and sufficient condition for the existence of a supervisory policy that enforces liveness. Additionally, we showed that a minimally restrictive supervisory policy that enforces liveness exists if and only if there is a supervisory policy that enforces liveness in a CtlPN. As a future research topic we suggest an investigation into improving the computational efficiency of the broad-brush procedures introduced in this paper.

Appendix

We present an algorithm that tests the existence of a path $\sigma = \sigma_1\sigma_2$ in the coverability graph $\hat{G}(N, \mathbf{m}^0)$ such that $v_0 \rightarrow \sigma_1 \rightarrow v_1 \rightarrow \sigma_2 \rightarrow v_1$, $\mathbf{x}(\sigma_2) > 0$ and $\mathbf{C}\mathbf{x}(\sigma_2) \geq 0$. First, we present some preliminaries from automata theory.

A *deterministic finite-state automation* (DFA) is a quintuple $P = (Q, \Sigma, \delta, s, F)$, where Q is the set of *states*, Σ is the *alphabet*, $s \in Q$ is the *initial state*, $F \subseteq Q$ is the set of *final states*, and δ: $Q \times \Sigma \rightarrow Q$ is the *transition function*. The transition function for strings, δ^*: $Q \times \Sigma^* \rightarrow Q$, is defined recursively as $\delta^*(q, \epsilon) = q$ and $\delta^*(q, \omega\sigma) = \delta(\delta^*(q, \omega), \sigma)$, where $\omega \in \Sigma^*$, $\sigma \in \Sigma$, and ϵ is the null-string. A string $\omega \in \Sigma^*$ is said to be *accepted* by P if and only if $\delta^*(s, \omega) \in F$. The set of all strings accepted by P is denoted as $L(P)$.

Letting T denote the set of transitions in a PN, the DFA $\hat{P} = (2^T, T, \hat{\delta}, \emptyset, T)$, with a state transition function $\hat{\delta}(\hat{T}, t) = \hat{T} \cup \{t\}$, where $\hat{T} \subseteq T, t \in T$, can be shown such that $L(\hat{P}) = \{\sigma \in T^* | \mathbf{x}(\sigma) > 0\}$.

The coverability graph of a PN N, $\hat{G}(N, \mathbf{m}^0) = (\hat{V}, \hat{A}, \hat{\Psi})$ can also be interpreted as a DFA. For each $v \in \hat{V}$ define a

[5]The class of 1-conservative PN's is a restricted class of bounded PN's, where for each transition the number of input places is equal to the number of output places.

[6]The complexity hierarchy satisfies the following containment: $NP \subseteq PSPACE \subset EXPSPACE$ (cf. [6, Th. 7.15, pp. 182–185]).

DFA $\tilde{P}(v) = (\hat{V}, T, \tilde{\delta}, v, \hat{V})$, where $\forall v_1 \in \hat{V}, \forall t \in T$

$$\tilde{\delta}(v_1, t) = \begin{cases} v_2, & \text{if } \exists a \in \hat{A} \text{ s.t. } \Psi(a) = (v_1, v_2) \\ & \text{and edge } a \text{ is assigned transition} \\ & t \text{ in } \hat{G}(N, \mathbf{m}^0). \\ \text{undefined}, & \text{otherwise.} \end{cases}$$

From the DFA's $\hat{P}$ and $\tilde{P}(v)$, we construct a DFA $\overline{P}(v) = (\hat{V} \times 2^T, T, \delta, (v, \emptyset), \{(v, T)\})$, where $\forall (v_1, \hat{T}) \in \hat{V} \times T, \forall t \in T$

$$\delta((v_1, \hat{T}), t) = \begin{cases} (\tilde{\delta}(v_1, t), \hat{T} \cup \{t\}), & \text{if } \tilde{\delta}(v_1, t) \text{ is defined} \\ \text{undefined}, & \text{otherwise.} \end{cases}$$

It can be shown that $L(\overline{P}(v)) = L(\hat{P}) \cap L(\tilde{P}(v))$. So, if there exists a path σ from the initial state $(v, \emptyset)$ to the final state (v, T), we know that $\mathbf{x}(\sigma) > 0$. Additionally, we can also infer that $v \to \sigma \to v$ in $G(N, \mathbf{m}^0)$. We call attention to the fact that in the DFA $\overline{P}(v)$, the state $(v, \emptyset)$ is not accessible from any members of the set $\hat{V} \times 2^T - \{(v, \emptyset)\}$.

In the graphical representation of $\overline{P}(v)$ (cf. [12, Fig. 2.2, p. 53] for additional details), each edge has a transition associated with it. Each transition has a column of the incidence matrix $\mathbf{C}$ associated with it. So, essentially each edge in the graphical representation of $\overline{P}(v)$ has an m-dimensional vector in $\{-1, 0, 1\}^m$ associated with it. We are concerned with the existence of a path σ from $(v, \emptyset)$ to (v, T) in the graphical representation of $\overline{P}(v)$ such that $\mathbf{Cx}(\sigma) \geq 0$. Before we present the details of this test, we review Kosaraju and Sullivan's polynomial time algorithm for detecting cycles in digraphs where edges are labeled using m-dimensional vectors [11].

Let $G = (V, A, \Psi)$ be a digraph and let Δ: $A \to \{-1, 0, 1\}^m$ be a function that assigns an m-dimensional vector to each edge, where each component of the vector belongs to the set $\{-1, 0, 1\}$. Let us suppose we are interested in the existence of some vertex v and a sequence of edges $\sigma = a_1 a_2 \cdots a_q$ in G such that $v \to \sigma \to v$ and $\sum_{i=1}^{q} \Delta(a_i) = 0$. Kosaraju and Sullivan [11] show the above problem is equivalent to the nonemptiness of the feasible region defined by the following inequalities in the set of real-valued variables $\{x_a\}_{a \in A}$:

1) $\forall v \in V, \sum_{a \in {}^\bullet v} x_a = \sum_{a \in v^\bullet} x_a$ (i.e., balancing condition for all vertices in the digraph);
2) $\forall a \in A, x_a \geq 0$;
3) $\sum_{a \in A} x_a \Delta(a) = 0$;
4) $\sum_{a \in A} x_a \geq 1$ (i.e., at least one edge is traversed)

where ${}^\bullet v = \{a \in A | \exists \hat{v} \text{ such that } \Psi(a) = (\hat{v}, v)\}$ and $v^\bullet = \{a \in A | \exists \hat{v} \text{ such that } \Psi(a) = (v, \hat{v})\}$. From an implementation viewpoint, these linear inequalities can be expressed as a feasible set of a linear programming problem while using some arbitrary linear cost function. The linear programming problem is infeasible if and only if the above set of inequalities define an empty set. The necessity of the observation follows directly from the interpretation of x_a as the number of traversals of edge a. To establish sufficiency, we note that the existence of a solution to the inequalities implies the existence of a rational solution. After suitable scaling, an integral-solution can be obtained. Using the interpretation of x_a as suggested above, the fact that there exists a v, σ, as required, can be established. We refer the reader to [11] for the details.

By replacing the third inequality (i.e., $\sum_{a \in A} x_a \Delta(a) = 0$) by $\sum_{a \in A} x_a \Delta(a) \geq 0$, we obtain a polynomial time algorithm for the existence of a $v \in V, \sigma = a_1 a_2 \cdots a_q$, such that $v \to \sigma \to v$ and $\sum_{i=1}^{q} \Delta(a_i) \geq 0$. However, we need a test for the existence of a path σ from $(v, \emptyset)$ to (v, T) in the graphical representation of $\overline{P}(v)$ such that $\mathbf{Cx}(\sigma) \geq 0$. We also know that $(v, \emptyset)$ is not accessible from any member of $\hat{V} \times 2^T - \{(v, \emptyset)\}$. So, if we add a new edge a_{new}, where $\Delta(a_{new}) = \mathbf{0}$, that originates from (v, T) and terminates at $(v, \emptyset)$ in the graphical representation of $\overline{P}(v)$, and if we added the additional constraint that $x_{a_{new}} > 0$, we obtain a polynomial time test for the existence of a path σ from $(v, \emptyset)$ to (v, T) such that $\mathbf{Cx}(\sigma) \geq 0$. Additionally, by construction of $\overline{P}(v)$, we know if such a σ exists, then $\mathbf{x}(\sigma) > 0$, and there exists a path in the coverability graph $\hat{G}(N, \mathbf{m}^0)$ such that $v \to \sigma \to v$. Also, by the coverability graph's construction, we are guaranteed that v is connected to the root vertex v_0. Since the number of vertices in the coverability graph is finite, the existence of σ can be decided in finite time.

ACKNOWLEDGMENT

The author would like to thank Prof. V. Chandru at the Indian Institute of Science, Bangalore, India, for bringing [11] to his attention.

REFERENCES

[1] Z. Banaszak and B. H. Krogh, "Deadlock avoidance in flexible manufacturing systems with concurrently competing process flows," *IEEE Trans. Robotics Automation*, vol. 6, pp. 724–734, Dec. 1990.
[2] M. Ben-Ari, *Principles of Concurrent Programming.* Englewood Cliffs, NJ: Prentice-Hall, 1982.
[3] C. G. Cassandras, *Discrete Event Systems: Modeling and Performance Analysis.* Homewood, IL: Irwin, 1993.
[4] S.-L. Chung, S. Lafortune, and F. Lin, "Limited lookahead policies in supervisory control of discrete event systems," *IEEE Trans. Automat. Contr.*, vol. 37, pp. 1921–1935, Dec. 1992.
[5] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms.* Cambridge, MA: MIT Press, 1990.
[6] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness.* New York: Freeman, 1979.
[7] Y.-C. Ho and X. Cao, *Perturbation Analysis of Discrete Event Dynamic Systems.* MA: Kluwer, 1991.
[8] L. E. Holloway and B. H. Krogh, "Synthesis of feedback control logic for a class of controlled petri nets," *IEEE Trans. Automat. Contr.*, vol. 35, pp. 514–523, May 1990.
[9] ——, "On closed-loop liveness of discrete-event systems under maximally permissive control," *IEEE Trans. Automat. Contr.*, vol. 37, pp. 692–697, May 1992.
[10] N. D. Jones, L. H. Landweber, and Y. E. Lien, "Complexity of some problems in Petri nets," *Theoretical Computer Sci.*, vol. 4, pp. 277–299, 1977.
[11] S. R. Kosaraju and G. F. Sullivan, "Detecting cycles in dynamic graphs in polynomial time," in *Proc. 20th Annu. ACM Symp. Theory Computing*, Chicago, IL, May 1988, pp. 398–406.
[12] H. R. Lewis and C. H. Papadimtriou, *Elements of the Theory of Computation.* Englewood Cliffs, NJ: Prentice-Hall, 1981.
[13] T. Murata, "Petri nets: Properties, analysis and applications," *Proc. IEEE*, vol. 77, no. 4, pp. 541–580, Apr. 1989.
[14] J. L. Peterson, *Petri Net Theory and the Modeling of Systems.* Englewood Cliffs, NJ: Prentice-Hall, 1981.
[15] P. J. Ramadge and W. M. Wonham, "Modular feedback logic for discrete event systems," *SIAM J. Contr. Optimization*, vol. 25, no. 5, pp. 1202–1218, Sept. 1987.

[16] _____, "Supervisory control of class of discrete event processes," *SIAM J. Contr. Optimization*, vol. 25, no. 1, pp. 206–230, Jan. 1987.
[17] C. Reutenauer, *The Mathematics of Petri Nets.* Hertforshire, U.K.: Masson and Prentice-Hall Int., 1990.
[18] G. S. Shedler, *Regenerative Stochastic Simulation.* Boston, MA: Academic, 1993.
[19] I. Suzuki and T. Murata, "A method for stepwise refinement and abstraction of Petri nets," *J. Computer Syst. Sci.*, vol. 27, pp. 51–76, 1983.
[20] A. Tanenbaum, *Operating Systems: Design and Implementation.* Englewood Cliffs, NJ: Prentice Hall, 1987.
[21] P. M. Vaidya, "An algorithm for linear programming which requires $O(((m+n)n^2+(m+n)^{1.5}n)L)$ arithmetic operations," in *Proc. 19th Annu. ACM Symp. Theory Computing*, New York, NY, May 1987, pp. 29–38.
[22] N. Vishwanadham, Y. Narahari, and T. L. Johnson, "Deadlock avoidance in flexible manufacturing systems using Petri net models," *IEEE Trans. Robotics Automation*, vol. 6, pp. 713–724, Dec. 1990.

Ramavarapu S. Sreenivas (S'83–M'93) received the B.Tech degree in electrical engineering from the Indian Institute of Technology, Madras, in 1985 and the M.S. and Ph.D. degrees in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, in 1987 and 1990, respectively.

He was a Post-Doctoral Fellow in decision and control at the Division of Applied Sciences, Harvard University, Cambridge, MA, before he joined the University of Illinois at Urbana-Champaign in September 1992.

Dr. Sreenivas is currently an Assistant Professor of General Engineering and a Research Assistant Professor with the Coordinated Science Laboratory. He is a recipient of the NSF Research Initiation Award. His research interests include modeling, analysis, control, and performance evaluation of discrete-state/discrete-event systems.