DES Workshop PN 2005
Miami, FL, June 21, 2005

A Survey on the Supervision of Petri Nets

Marian Iordache[1] and Panos Antsaklis[2]

[1] LeTourneau University, Longview, TX 75607-7001, USA,
MarianIordache@letu.edu,
WWW home page: http://www.letu.edu/people/marianiordache
[2] University of Notre Dame, Notre Dame, IN 46556, USA
antsaklis.1@nd.edu,
WWW home page: http://www.nd.edu/~pantsakl

Abstract. The paper focuses on supervision approaches in which the supervisor is a Petri net (PN). Special attention is given to the monitor-based approaches. In monitor-based supervision, the set of transitions (events) of a supervisor is a subset of the set of transitions (events) of the plant. The significance of monitor-based approaches to the design of the general PN supervisors is emphasized. The approaches surveyed in the paper are classified based on the type specifications, controllability/observability assumptions, and centralized/decentralized type of supervision. Included are also the literature results that do not lead to a supervisor represented as a PN.

1 Introduction

This paper contains a survey of supervision methods used in the literature, with a focus on the methods in which the supervisor is modeled by a Petri net (PN). The PN variant considered in this paper are the P/T nets. The basic PN model is extended, when needed, to account for uncontrollability/unobservability and transition labels. The basic PN model has a structure $\mathcal{N}$ defined by $\mathcal{N} = (P, T, D^-, D^+)$, where P is the set of places, T the set of transitions, $D^-, D^+ \in \mathbb{N}^{|P|\times|T|}$ are the input and output matrices, and $\mathbb{N}$ is the set of nonnegative integers. We denote by $D = D^+ - D^-$ the incidence matrix and by μ the marking. A PN with initial marking μ_0 will be denoted by $(\mathcal{N}, \mu_0)$.

In monitor-based supervision, a plant PN is enhanced with additional places, called **monitors**, such that given specification constraints are satisfied. One of the inherent qualities of this approach is that the operation of the plant under supervision can also be modeled by a PN, which we call **closed-loop PN**. The closed-loop PN is simply the plant plus the monitor places.

The most common type of specifications used with monitor-based supervision has the form

$$L\mu \leq b \tag{1}$$

where μ is the marking, $L \in \mathbb{Z}^{n_c\times m}$, $b \in \mathbb{Z}^{n_c}$, $\mathbb{Z}$ is the set of integers, m is the number of places of the PN, and n_c the number of constraints. The constraints

(1) are also known as *generalized mutual exclusion constraints* [1], since a simpler form of (1) corresponds to mutual exclusion specifications.

The enforcement of marking constraints is overviewed in section 2. Note that the focus is on the linear marking constraints (1), though methods that can deal with more (or less) general marking specifications are also surveyed. The constraints (1) have been used in the context of a constrained optimal control problem of chemical processes [2], for the coordination of AGVs [3], as manufacturing constraints in [4], and for mutual exclusion in batch processing [5]. An extension of this type of constraints has also been used for the supervision of railway networks [6].

The constraints used in [6] belong to the class of generalized linear constraints [7], which we describe in section 3. These can still be enforced by monitors, as monitors can enforce constraints more general than (1). Generalized linear constraints are important because they describe the class of constraints that can be enforced by monitors.

More general specifications can be described by languages or as disjunctions of constraints. The enforcement of P-type languages is considered in section 4. Note that a P-type language is a language that describes the firing sequences of a labeled PN with an initial marking [8]. Note also that in section 4 the plant is modeled by labeled PNs. The enforcement of P-type languages in the context of free-labeled PNs (i.e. PNs with distinct labels), corresponds to the enforcement of generalized linear constraints described in section 3.

Disjunctions of constraints (1) have the form $L_1\mu \leq b_1 \vee L_2\mu \leq b_2 \vee \ldots L_p\mu \leq b_p$, requiring the marking μ to satisfy at least one of $L_i\mu \leq b_i$, for $i = 1 \ldots p$. The enforcement of disjunctions of constraints (1) is considered in section 5. Unlike to the previous types of constraints, which typically result in monitor-based supervisors, in this case we need the more powerful supervisors in which the set of transitions (events) is not necessarily a subset of the set of transitions (events) of the plant PN.

Decentralized supervision is covered in section 6. In decentralized supervision, the supervisor consists of components that act on subsystems of the plant such that a global specification is satisfied. Several decentralization settings could be distinguished depending on the type of communication between the supervisor components. The section surveys an extension from centralized to decentralized supervision of the monitor-based design. Note that the extension applies to supervision settings with no communication, restricted communication, and unlimited communication.

This paper is an abbreviated version of the technical report [9]. The reader is referred to [9] for additional information.

2 Marking Constraints

We first consider approaches for the linear marking constraints (1). This type of constraints have been shown to have the following properties. They can describe any forbidden marking specification on safe Petri nets [10, 1], where a Petri net

is *safe* if all reachable markings are binary vectors (i.e. consisting of 0 and 1 elements). This property is very interesting for supervision problems on certain subclasses of Petri nets, and notably on marked graphs. Further, as shown in the next sections of the paper, more general specifications can be reduced to specifications (1) on transformed PNs. Thus, the methods developed for the enforcement of (1) are relevant for a much wider class of constraints, including language specifications and disjunctive constraints. Finally, the constraints (1) are also important for the representation of deadlock prevention and liveness specifications [11, 12].

In the fully controllable and observable case, the construction of a supervisor enforcing (1) is as follows. Given a PN $(\mathcal{N}, \mu_0)$ of structure $\mathcal{N} = (P, T, D^-, D^+)$, the supervisor is described by $(\mathcal{N}_s, \mu_{0,s})$ with $\mathcal{N}_s = (P_s, T, D_s^-, D_s^+)$, where

$$D_s = -LD \tag{2}$$

$$\mu_{0,s} = b - L\mu_0 \tag{3}$$

Note that the places of the supervisor are called **monitors**. An example is shown in Figure 1(b), in which the monitors p_7 and p_8, in this order, correspond to the constraints

$$\mu_1 + \mu_2 + \mu_5 \leq 1 \tag{4}$$

$$\mu_3 + \mu_7 \leq 1 \tag{5}$$

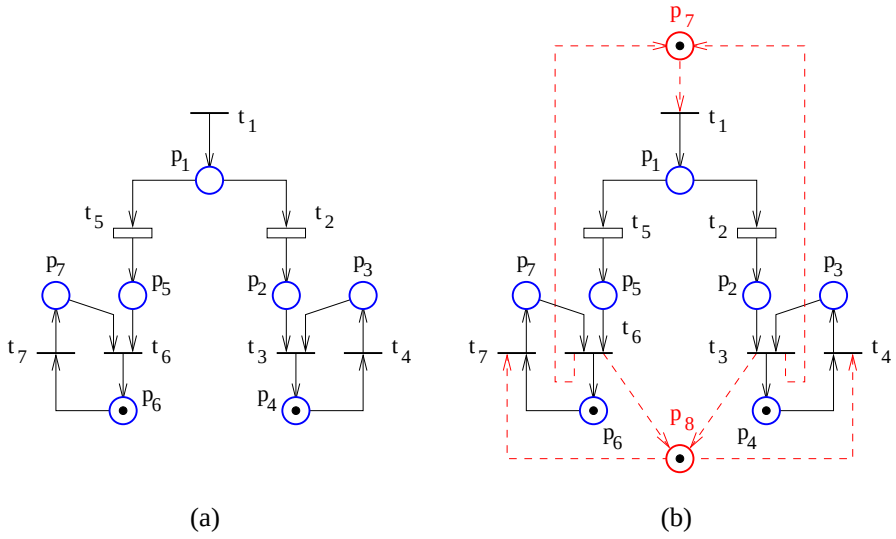


Fig. 1.

The method presented above is known as *supervision based on place invariants* and has been detailed in a number of references, such as [1, 10, 4, 13]. While in the early papers the elements of L have been assumed nonnegative, note that the method applies for any integer matrices L and b.

The literature methods for the enforcement of the linear marking constraints can be distinguished based on the approach they follow when the plant is not fully controllable and/or observable. First, the methods relying on the concept of *admissibility* will be discussed.

A supervisor is **admissible**, when it respects the uncontrollability and unobservability constraints of the plant. The constraints $L\mu \leq b$ are **admissible** if the supervisor defined by (2–3) is admissible. In an admissibility-based approach, the constraints $L\mu \leq b$ are transformed (whenever possible) to an admissible form $L_a\mu \leq b_a$ such that

$$L_a\mu \leq b_a \Rightarrow L\mu \leq b \tag{6}$$

Then, the supervisor enforcing $L_a\mu \leq b_a$ is admissible, and enforces $L\mu \leq b$ as well. Instead of transforming our constraints to the form $L_a\mu \leq b_a$, we could use the (more general) disjunctive form $\bigvee_i L_{a,i}\mu \leq b_{a,i}$. Then, the approach of section 5 could be used to obtain a PN representation of the supervisor.

The computation of the sets $L_a\mu \leq b_a$ can be simplified when we express the admissibility requirement by linear inequalities. Such sufficient conditions for admissibility can be found under various settings [9]. In the most common setting of monitor-based supervision, transitions are classified as controllable (observable) and uncontrollable (unobservable). Controllable (observable) transitions can be individually controlled (observed). Let T_{uc} and T_{uo} be the sets of uncontrollable and unobservable transitions of the plant. The following conditions are sufficient to ensure the admissibility of (1)

$$LD(\cdot, T_{uc}) \leq 0 \tag{7}$$

$$LD(\cdot, T_{uo}) = 0 \tag{8}$$

These conditions require that a monitor should not be an input to an uncontrollable transition and should not be connected to an unobservable transition, except by input and output arcs of equal weight. As an example, consider Figure 1(a). Assuming t_2 and t_5 uncontrollable, $\mu_2 + \mu_5 \leq 1$ is not admissible, as enforcing it may attempt controlling either of t_2 and t_5. However, it can be checked that $\mu_1 + \mu_2 + \mu_5 \leq 1$ is admissible and $\mu_1 + \mu_2 + \mu_5 \leq 1 \Rightarrow \mu_2 + \mu_5 \leq 1$.

Another interesting setting is the one of labeled PNs. Given a labeling function $\rho : T \to \Sigma \cup \{\lambda\}$, where λ is the null event, we can write sufficient conditions similar to (7–8):

$$\forall t_1, t_2 \in T,\ \rho(t_1) = \rho(t_2) \Rightarrow LD(\cdot, t_1) = LD(\cdot, t_2) \tag{9}$$

$$\forall t \in T,\ \rho(t) \in \Sigma_{uc} \cup \{\lambda\} \Rightarrow LD(\cdot, t) \leq 0 \tag{10}$$

$$\forall t \in T,\ \rho(t) \in \Sigma_{uo} \cup \{\lambda\} \Rightarrow LD(\cdot, t) = 0 \tag{11}$$

Note that relation (9) constrains a monitor place to be identically connected to any transitions t_1 and t_2 that have the same label. An important observation is

that since (9–11) have the same form as (7–8), any methods based on (7–8) can be adapted to labeled PNs.

The design of admissible constraints has been approached in [4, 13] using the following parameterization:

$$L_a = R_1 + R_2 L \tag{12}$$

$$b_a = R_2(b + \mathbf{1}) - \mathbf{1} \tag{13}$$

R_1 is an integer matrix with nonnegative elements and R_2 is a diagonal matrix with positive integers on the diagonal. This parameterization is used as a sufficient condition for (6). Now, the problem is to find L_a and b_a subject to (12–13), (7) and (8). This is a linear integer programming problem for which, sometimes, solutions may be found using an efficient matrix row operation algorithm [4, 13]. Note that this integer programming formulation of the problem allows introducing additional requirements of interest. For instance, communication constraints and a minimum-communication objective were used in a distributed version of this problem [14]. Note that the approach of [4, 13] is suboptimal. That is, a solution may not be found when solutions do exist, and if one is found, it may not be the least restrictive solution. A source of suboptimality is that the computation is not constrained to ensure that if L'_a and b'_a are another solution to (12–13), (7) and (8), then $L_a\mu \leq b_a \not\Rightarrow L'_a\mu \leq b'_a$.

The approach of [4, 13] can been improved in several ways. First, it should be noticed that it is difficult to express by linear inequalities the requirement that $L_a\mu \leq b_a$ should be as permissive as possible. However, it is easy to constrain the computation of L_a and b_a to guarantee some weaker properties: (a) that a set of markings of interest is included in $\{\mu : L_a\mu \leq b_a\}$ and (b) that a set of firing count vectors x is included in $\{x : D_c x \geq 0\}$, where D_c is the incidence matrix of the closed-loop. These simple extensions can be found in [14]. As noticed in [15], the admissible constraints $L_a\mu \leq b_a$ satisfying (6) may not have a unique supremal element. Thus, further work has been done by the authors of [16] towards finding the supremal constraints $L_a\mu \leq b_a$ subject to (12–13), (7) and (8) by means of a parameterization.

Another way to control the selection of L_a and b_a is by means of observation and control costs. Thus, in [17], the optimal design of supervisors is considered, where optimality here is with respect to control and observation costs. Here, instead of having sets of uncontrollable and unobservable transitions T_{uc} and T_{uo}, we have maps $z_c : T \to \mathbb{R}^+$ and $z_o : T \to \mathbb{R}^+$, associating control and observation costs to each transition. The setting of [17] is general, as we can still consider some transitions as uncontrollable/unobservable by associating with them very large control or observation costs. The design problem of [17] is solved by an integer programming approach, using (12–13) and admissibility conditions equivalent to (7) and (8).

The optimal design of supervisors with respect to the admissibility constraints (7) and (8) is approached also in chapter 8 of [18]. The proposed method applies to specifications (1) in which for all rows of L, all elements on a row have

the same sign. Note that the solution is given in the form of a disjunction of constraints.

Still another approach appears in [19]. The setting of [19] assumes full observability. Essentially, given the constraint $l\mu \leq c$ with $l \in \mathbb{N}^m$ and $c \in \mathbb{N}$, $l\mu \leq c$ is replaced with the disjunction

$$\bigvee_{l_i \in SD_{min}(l)} [l_i\mu \leq c] \tag{14}$$

where $SD_{min}(l)$ is the set of minimal integer vectors x satisfying $x \geq l$ and $xD(\cdot, T_{uc}) \leq 0$. In particular, $l\mu \leq c$ is replaced with the single admissible constraint $l_1\mu \leq c$ when $SD_{min}(l)$ is the singleton $\{l_1\}$. Under the conditions of [20, 21], which are discussed later in this section, the resulting supervisor is least restrictive. It is interesting to notice that some of the assumptions of [19] can be dropped. Indeed, (14) is still a valid supervisor even if $l \in \mathbb{Z}^m$ and $c \in \mathbb{Z}$ (as opposed to $l \in \mathbb{N}^m$ and $c \in \mathbb{N}$). Further, partial observability can be incorporated by defining $SD_{min}(l)$ as the set of minimal integer vectors x satisfying $x \geq l$, $xD(\cdot, T_{uc}) \leq 0$ and $xD(\cdot, T_{uo}) = 0$.

Another class of literature results deal with the calculation of the *maximal controlled-invariant set*, assuming full observability. Let $\mathcal{M}_F$ be the set of forbidden markings and $\mathcal{N}_u = (P, T_{uc}, D^-(\cdot, T_{uc}), D^+(\cdot, T_{uc}))$ a subnet of the plant $\mathcal{N}$ that does not contain the controllable transitions. The *maximal controlled-invariant set* [22, 3] is defined as

$$\mathcal{A}_F = \{\mu : \mathcal{R}(\mathcal{N}_u, \mu) \cap \mathcal{M}_F = \emptyset\} \tag{15}$$

It represents the set of markings (states) from which $\mathcal{M}_F$ can be avoided. Once we know $\mathcal{A}_F$, the control task is simply to disable any control actions that lead to a marking in $\mathcal{A}_F$. Note that avoiding $\mathcal{A}_F$, as opposed to some superset $\mathcal{E} \supseteq \mathcal{A}_F$, corresponds to least restrictive supervision. In particular, as noticed in [1], solutions replacing a specification $L\mu \leq b$ with an admissible $L_a\mu \leq b_a$ correspond to supervisors that avoid supersets $\mathcal{E} \supseteq \mathcal{A}_F$, since $\mathcal{A}_F$ may not be representable as the complement of a set of constraints of the form (1) even when $\mathcal{M}_F$ is given as the complement of a set of constraints (1).

In [21] specifications (1) are considered, where L and b are restricted to have only nonnegative elements. Given $l\mu \leq c$ as one of the constraints of (1), (so $l \in \mathbb{N}^m$ and $c \in \mathbb{N}$), the influential subnet $\mathcal{N}_u^l$ is defined, which is the subnet of $\mathcal{N}_u$ containing the places p with $l(p) \neq 0$ and the directed paths of $\mathcal{N}_u$ to these places. The main result of the paper shows how to express $\mathcal{A}_F$ as the set of markings satisfying a disjunction of linear marking inequalities. This result relies on two conditions, as follows. First, $\mathcal{N}_u^l$ should be a marked graph. (Note that $\mathcal{N}_u^l$, not $\mathcal{N}$, is restricted to a marked graph structure.) Second, for all reachable markings of $(\mathcal{N}, \mu_0)$, every directed circuit of $\mathcal{N}_u^l$ should have at least one token. In [21] the supervisor is not represented as a PN. However, the subsequent work of [19] proposes an extended PN representation of the supervisor, in which negative markings are allowed. Note that a similar result was obtained in [20] for the case in which $\mathcal{N}_u^l$ is a state machine, instead of a marked graph. For this case,

it is shown that $\mathcal{A}_F$ has the form $\mathcal{A}_F = \{\mu : l_a\mu \leq c\}$, where l_a can be easily computed. Thus, the monitor enforcing $l_a\mu \leq c$ is the least restrictive supervisor.

The efficient computation of [20] for PNs and specifications for which the subnets $\mathcal{N}_u^l$ are state machines, may not be surprising in light of the complexity findings of [23]. The model of [23] is as follows. The plant consists of p components that do not interact with each other, where the components are represented by deterministic Büchi automata $G_i = (Q_i, \Sigma_i, \delta_i, q_{0i}, Q_{mi})$ over disjoint alphabets Σ_i. Given the subsets of states $\overline{Q}_i \subset Q_i$, a mutual exclusion specification requires less than k components to have their states q_i in $\overline{Q}_i$ at the same time. Note that the plant of [23] can be represented by a safe labeled PN with a state machine structure, and the mutual exclusion constraint by a constraint $l\mu \leq c$ in which $c = k$ and all elements of l are 0 or 1. One of the problems considered in [23] is to find nonblocking coordinators that enforce the mutual exclusion constraint. Roughly, a nonblocking coordinator is a supervisor that guarantees certain strong liveness properties. The paper shows that the existence of a solution can be decided in polynomial time in p and n, where $n = \max_i |Q_i|$. Further, it is shown that if a solution exists, the minimally restrictive solution can be found in polynomial time in p and n. It is interesting to note that in the equivalent PN representation of the plant, the supervisor found in [23] corresponds to a monitor place enforcing a constraint $l_a\mu \leq c$, provided the PN is free-labeled. Note also that in view of [24], the assumption that the sets Σ_i are disjoint seems to be critical for polynomial complexity. In [24] it is shown that when the components of the plant have a shared event, the solvability of the problem can no longer be decided in polynomial time. A restriction of the problem for which polynomial complexity is maintained is also proposed.

A method that finds the optimal design for specifications (1) appears in [25]. Several assumptions are made, as seen from the following outline of the method. Let $\mathcal{L}(\mathcal{N}_u, \mu)$ denote the set of firing sequences σ of $\mathcal{N}_u$ that are enabled at the marking μ. Let $\underline{\sigma}$ be the firing count vector with respect to $\mathcal{N}$ (not $\mathcal{N}_u$). Finally, let $l\mu \leq c$, $l \in \mathbb{Z}^{|P|}$ and $c \in \mathbb{Z}$, denote a single constraint of (1). The set $\mathcal{A}_F$ corresponding to $l\mu \leq c$ is given by $\mathcal{A}_F = \{\mu : (\forall \sigma \in \mathcal{L}(\mathcal{N}_u, \mu))\, l\mu + lD\underline{\sigma} \leq c\}$. By assuming $\mathcal{N}_u$ (not $\mathcal{N}$) to be acyclic, $\mathcal{A}_F = \{\mu : l\mu + lDv^*(\mu) \leq c\}$, where $v^*(\mu)$ is the solution of the linear integer program $\max lDv$ subject to $D(\cdot, T_{uc})v \geq -\mu$ and $v \geq 0$. As shown in [25], a closed-form expression of $\mathcal{A}_F$ can be computed under additional assumptions. First, [25] defines subnets for each $t \in T_{uc}$, consisting of all paths of $\mathcal{N}_u$ ending in t. Denoting by $\hat{T}_{uc} = \{t \in T_{uc} : lD(\cdot, t) > 0\}$, [25] requires all subnets of $t \in \hat{T}_{uc}$ be independent (disjoint). Further, when the subnets have the $TS1$ structure described in [25], $\mathcal{A}_F$ can be expressed by a disjunction of inequalities: $\mathcal{A}_F = \{\mu : \bigvee_{i=1}^{k} l_i\mu \leq c\}$ for some k and $l_i \in \mathbb{Z}^{|P|}$. Moreover, when the subnets have the $TS2$ structure described in [25], then $\mathcal{A}_F = \{\mu : l_a\mu \leq c\}$ for some $l_a \in \mathbb{Z}^{|P|}$. Thus, in the $TS1$ case the optimal supervisor of $l\mu \leq c$ enforces $\bigvee_{i=1}^{k} l_i\mu \leq c$, and in the $TS2$ case $l_a\mu \leq c$. The approach of [25] is computationally efficient, as $\mathcal{A}_F$ is calculated independently of μ and without resorting to the traditional methods for solving integer programs.

Results on the supervision of marked graphs appear in [26]. Compared to [22, 3], the marked graphs considered here may not be safe. However, unlike to [22, 3], the results are presented in the no concurrency setting and the uncontrollability model is simpler: the set of transitions is partitioned into controllable (T_c) and uncontrollable (T_{uc}) transitions. The specifications have the form (1). A least restrictive supervision policy is computed first for several particular cases. This policy is very efficient, as it involves little online computations. Finally, a supervision policy is proposed for the general case, which involves solving online linear programs, for every reachable marking. This last result is based on the observation that given a constraint $l\mu \leq c$, $l \in \mathbb{Z}^{1\times m}$ and $b \in \mathbb{Z}$, finding $\max\{l\mu^* : \mu^* \in \mathcal{R}(\mathcal{N}_u, \mu)\}$ is equivalent to the integer linear program $\max\{l\mu^* : \mu^* = \mu + D(\cdot, T_{uc})q,\ q \in \mathbb{N}^{|T_{uc}|}\}$, which is equivalent to the linear program $\max\{l\mu^* : \mu^* = \mu + D(\cdot, T_{uc})q,\ q \in \mathbb{R}_+^{|T_{uc}|}\}$. These two equivalences result from the fact that the plant is a live marked graph.

In [27], the supervisory control problem is approached based on the reachability graph. Here, the supervisor is designed as a set of monitors acting upon the PN plant. First, a subset of the reachability graph is obtained, such that from any of the markings of the subgraph, forbidden states and blocking states cannot be reached by firing uncontrollable transitions. This subgraph becomes the desired reachability graph that is to be achieved by the closed-loop. Then, the authors deal with the design of supervisors that ensure the closed-loop has the specified reachability graph. Given a set Ω containing the pairs (μ, t) such that t should be disabled at the marking μ, monitors are designed, such that each monitor deals with at least one of the pairs (μ, t) of Ω. The connections of a monitor to the plant are determined by finding an integer solution to a system of inequalities. Due to the particular form of the inequalities, the solution can be found using linear programming.

Several important results on the control of live marked graphs appear in [28]. The specification considered there is more powerful than (1), as it has the form $av \leq c$, where v is the Parikh vector, $a \in \mathbb{Z}^{1\times n}$ and $c \in \mathbb{Z}$. In [28], the set of transitions T is partitioned into the disjoint subsets: $T = T_c \cup T_f \cup T_i$, where T_c is the set of controllable transitions, and $T_o = T_c \cup T_f$ the set of observable transitions. The approach of the paper is as follows. *Suspect* vectors are defined as Parikh vectors[3] v such that $v|_{T_o} = v'|_{T_o}$ for some v' with the property that after firing v', a forbidden state v'' could be reached (i.e. $av'' > c$), by firing only uncontrollable transitions. The paper shows that any deterministic supervisor has to avoid reaching the set of suspect vectors, and that the projections of these vectors on T_o form a convex set (that is, the set of integral points of a polyhedron). The paper shows also how to compute this set. Since the complement of this set may not be convex, it follows that the least restrictive supervisor may not be implementable by control (monitor) places. Even when monitors can be used, the paper shows that the number of monitors may be exponential. Another observation of the authors is that the number of linear constraints defining the set of suspect vectors may depend exponentially on the size of $D(\cdot, T_{uc})$. The

[3] A definition can be found in section 3 for the definition.

alternative to the computation of this set is as follows. Given a state v_0, a linear program can be solved in order to decide whether $t \in T_c$ should be enabled. Since linear (not integer linear) programming is used, the computation has polynomial complexity.

A different plant model used in the literature is the controlled PN (CtlPN). In CtlPNs, a supervisor is viewed as a (possibly nondeterministic) function that determines the controls to be applied to the plant, based on the observation of the plant. Thus, the supervisor and the closed-loop are not represented as PNs. The main distinguishing feature of CtlPNs is that a supervisor is given the ability to enable/disable groups of transitions, not individual transitions. This is a more difficult case which, in the context of PN supervisors, could be dealt with in the setting of double-labeled PNs [9]. On the other hand, supervision problems in which the transitions of the plant are individually controllable or uncontrollable can obviously be approached in the CtlPN setting.

Results on the supervision of CtlPNs were obtained first for CtlPNs with a marked graph structure in [22, 3]. The setting of [22, 3] is as follows. The plant is a CtlPN in which the underlying PN is a cyclic marked graph with an initial marking that places exactly one token in every directed cycle. Thus, the PN is safe (i.e., all reachable markings are binary vectors). Full observability is implicitly assumed. The supervisory goal is to avoid a set of forbidden markings $\mathcal{M}_F$. While in [22] $\mathcal{M}_F$ is specified in terms of *place*, *set* and *class conditions*, [3] specifies $\mathcal{M}_F$ as the complement of the feasible set a set of constraints (1):

$$\mathcal{M}_F = \bigcup_{(F,k)\in\mathcal{F}} \left\{ \mu : \sum_{p\in F} \mu(p) > k \right\} \tag{16}$$

Note that both *class conditions* and (16) can specify any set $\mathcal{M}_F$, due to the fact that the PN model is a safe cyclic marked graph. (It is possible to obtain inequalities (1), not (16) though, from any set $\mathcal{M}_F$ of a safe PN [10].) There are some mild assumptions on the set $\mathcal{M}_F$ in [22, 3]. As mentioned in [29], these assumptions guarantee the supervisor designed is least restrictive.

The design of supervisors in [22] is approached by analyzing the paths of the marked graph that do not involve controllable transitions. This solution is simplified in [3]. The solution of [3] involves the following: identify a number of paths in the marked graph, offline; evaluate certain place and path predicates, online.

In [30], the design of supervisors is studied for a setting similar to that of [22, 3], in which the CtlPN has a state machine structure. The forbidden set here is described by disjunctions of constraints of the form (1). The use of disjunctions is necessary in order to describe arbitrary sets of forbidden states, as the PN is not assumed to be safe.

The results of [22, 3] are generalized in [31], by extending the plant model from marked graphs to arbitrary ordinary PNs. The type of specifications is similar to that of [22]. However, as the PNs may not be safe, it cannot capture all possible sets of forbidden markings. Further, compared with (1), the specifications of [31] are neither a subset nor a superset of the specifications expressed

by (1). As in the previous work [22, 3], in [31] the least restrictive supervisor is found by a path based approach.

The extension of the path-based approach of Holloway and Krogh [22, 3] to partial observability appears in [32]. Ordinary PN structures are considered, instead of marked graphs. The set of transitions is partitioned in controlled and uncontrolled transitions, $T = T_c \cup T_{uc}$, and in observed and unobserved transitions, $T = T_o \cup T_{uo}$, with $T_o \supseteq T_c$. Note that a transition is controlled if connected to some control place. Further, each transition is labeled by one event, and a transition is observed if its label is not the null event. The authors propose a path algebra, described in more detail in [31]. This algebra is used to define reachability predicates, which are then used to define the least restrictive control policy. (The supervision is nondeterministic, so least restrictive control policies exist.)

3 Generalized Linear Constraints

Generalized linear constraints have the form

$$L\mu + Hq + Cv \leq b \tag{17}$$

where q is the firing vector and v the Parikh vector. To simplify our presentation, we will focus on the no concurrency assumption. The general case can be found in [33]. Under the no concurrency assumption, $q \in \{0,1\}^n$, $n = |T|$, identifies the transition that is to be fired next: $q_i = 1$ if t_i is to be fired next, and $q_i = 0$ otherwise. The Parikh vector $v \in \mathbb{N}^n$ records how many times each transition has fired. For instance, $v_1 = 4$ indicates t_1 has fired four times. q and v are illustrated in Figure 2. Further, $H \in \mathbb{Z}^{n_c \times n}$ and $C \in \mathbb{Z}^{n_c \times n}$ are matrices, and n_c is the number of constraints.

The constraints (17) are interpreted as follows. A supervisor enforcing (17) ensures that: (i) all states (μ, v) satisfy $L\mu + Cv \leq b$; (ii) if q is the firing vector of a transition t_i, $\mu \xrightarrow{t_i} \mu'$, and $v' = v + q$, then $L\mu + Hq + Cv \leq b$ and $L\mu' + Cv' \leq b$.

In [7] it is shown that:

- the class of constraints
$$Hq + Cv \leq b \tag{18}$$
is as general as the class $L\mu + Hq + Cv \leq b$. That is, given $L\mu + Hq + Cv \leq b$, there is C' such that $L\mu + Hq + Cv \leq b$ and $Hq + C'v \leq b$ are equivalent.
- any monitor arbitrarily connected to the places of a Petri net can be described as enforcing a constraint of the form (18), where b corresponds to the initial marking of the monitor.
- in fact, any PN $(\mathcal{N}, \mu_0)$, $\mathcal{N} = (P, T, D^-, D^+)$, can be described by constraints (18), for $H = D^-$, $C = D^- - D^+$ and $b = \mu_0$.
- Consequently, the specifications (17) correspond to the P-type languages of the free-labeled PNs! (Following [8], a labeled PN is freely-labeled when each

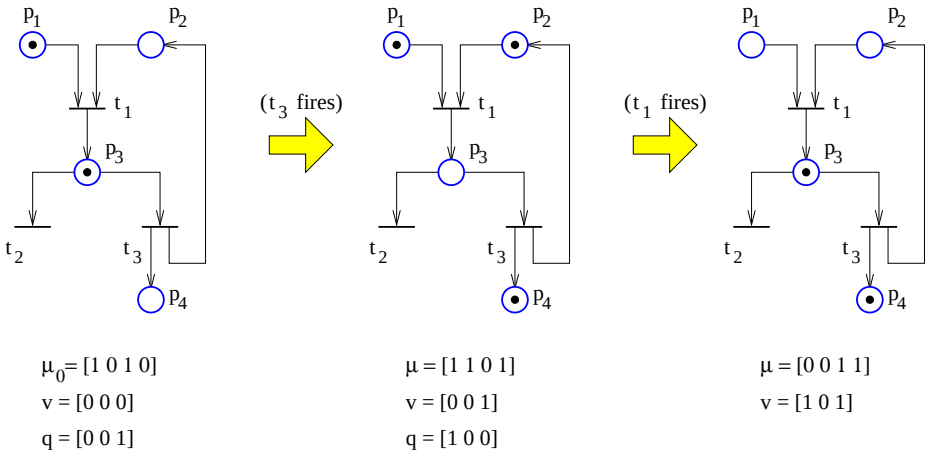


Fig. 2. Illustration of the q and v variables.

transition of the net has a unique and distinct label, different from λ, the null symbol; further, a language $\mathcal{L}$ is a P-type PN language if there is a PN with an initial marking such that $\mathcal{L}$ consists of the words associated with the firing sequences enabled by the initial marking.)

Let $\mathcal{L}$ be the language corresponding to all behaviors accepted by a specification (17). It is important to note that the specification (17) does not require the closed-loop to generate $\mathcal{L}$. Rather, it requires the closed-loop language to be a sublanguage of $\mathcal{L}$. Further, note that the least restrictive supervisor enforcing (17) can be easily designed under full controllability and observability assumptions [7, 33].

Another important result that appears in [7, 33] shows that the design of supervisors enforcing (17) can be reduced to the design of supervisors enforcing (1) [7, 33]. Thus, if (17) is to be enforced on a PN $\mathcal{N}$, the problem is transformed into the design of a supervisor enforcing constraints of the form (1) on a PN $\mathcal{N}_H$. The solution to this problem is then used to obtain the solution to the original problem of designing a supervisor enforcing (17) on $\mathcal{N}$. The uncontrollability and unobservability setting used in [7, 33] is that transitions are either individually controllable (observable) or uncontrollable (unobservable).

4 Language Constraints

As mentioned in section 3, it is possible to reduce the problem of enforcing the P-type languages of free-labeled PNs to the problem of enforcing constraints (1). This section shows that we can approach in a similar way more general problems, that do not assume free-labeling for the plant and the specification. As in the

previous considerations, the requirement here is that the closed-loop generates a sublanguage of the specification.

As an example, consider the PN and the specification shown in Figure 3. In this example, the specification is described by a PN labeled by the events a and b. To simplify the notation, it is assumed that all events of the plant that do not appear in the specification are always enabled in the specification. The closed-loop in our example can be computed immediately by a parallel composition of the plant and specification, and is shown in Figure 4(a). Note that in the closed-loop, the transition t_1 of the plant appears in the form of t_1^1 and t_1^2, corresponding to the synchronization of t_1 with the transitions t^1 and t^2 of the supervisor. Similarly, t_2^3 and t_2^4 correspond to the synchronization of t_2 with t_3 and t^4. A formal description of the algorithm composing PN plants with PN specifications can be found in [6].

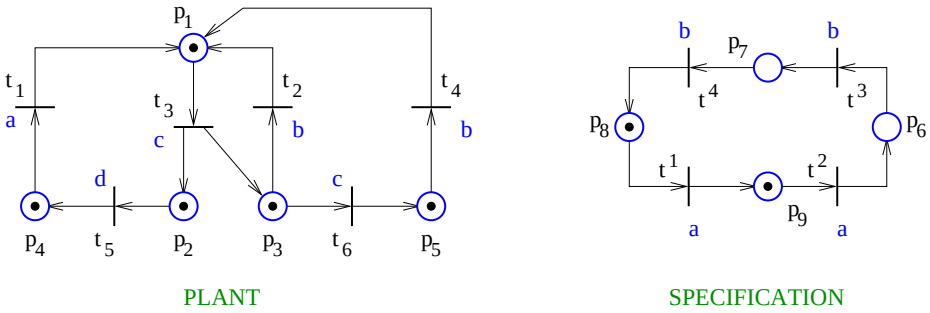


Fig. 3.

The supervision is interpreted as follows. The plant and the supervisor have each a distinct set of transitions, T_p and T_s, respectively. The supervisor cannot observe/control the plant transitions directly, but it can observe/control events generated by the plant. When the plant generates the event a, the supervisor picks one of its own enabled transitions $t \in T_s$ that is labeled by a, and fires it. Note that the supervisor is free to choose which of its enabled transitions labeled by a fires. For instance, in Figure 3, when the plant generates a, the supervisor can select either of t^1 or t^2, since both are enabled and labeled by a. So we can relabel the closed-loop, to indicate the supervisor can distinguish between its own transitions that have the same label. Thus, in Figure 4 we have the following new labels: a^1 for t_1^1, a^2 for t_1^2, b^3 for t_2^3 and t_4^3, and b^4 for t_2^4 and t_4^4.

According to our previous section 3, in the closed-loop, every place of the supervisor corresponds to a specification in terms of constraints (17). For instance, p_9 enforces $v_1^2 - v_1^1 \leq 1$ and p_8 enforces $v_1^1 - v_2^4 - v_4^4 \leq 1$. This gives us a readily

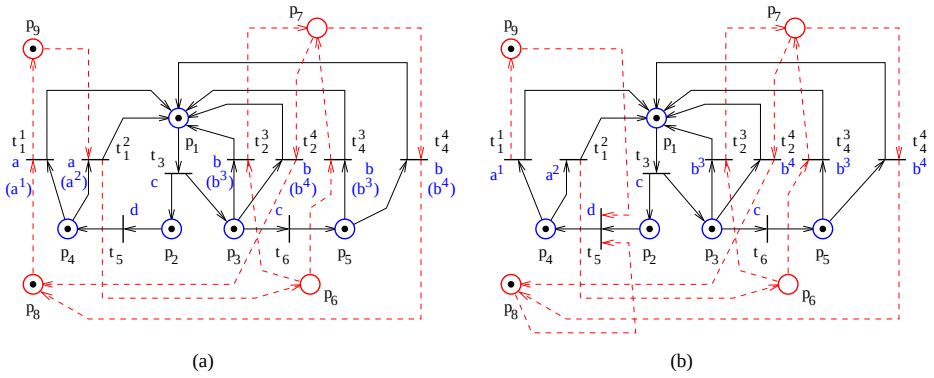


Fig. 4.

available approach for supervisor design in the case of partial controllability and partial observability:

- Compose the PN plant and the PN specification (supervisor).
- Relabel the closed-loop, to take in account the supervisor can distinguish between its own transitions.
- Find the constraints (17) corresponding to the constraints enforced by the monitors of the closed-loop.
- Transform the constraints (17) to an admissible form, which is at least as restrictive.

For instance, assume in our example that t_1 (the event a) is uncontrollable but the other transitions are controllable. Assume all other events are observable. Notice that in Figure 4(a) p_8 and p_9 may attempt disabling t_1. So, the specification is inadmissible. However, the constraints enforced by p_8 and p_9, namely $v_1^1 - v_2^4 - v_4^4 \leq 1$ and $v_1^2 - v_1^1 \leq 1$, can be transformed to the admissible form $v_1^1 - v_2^4 - v_4^4 + \mu_4 \leq 1$ and $v_1^2 - v_1^1 + \mu_4 \leq 1$. The resulting closed-loop and supervisor are shown in Figure 4(b) and Figure 5, respectively. The supervision is admissible, while ensuring the plant generates only words that satisfy the original specification of Figure 3.

It is known that the supremal controllable sublanguage of a P-type PN language may not be a P-type PN language [34]. This is an indication that the approach presented here is suboptimal, in the sense that it may not lead to the least restrictive supervisor. Note that in the literature it has been shown that the computation of the least restrictive supervisor can be reduced to a forbidden marking problem, provided both the plant and specification generate deterministic languages [35]. (Given a labeled PN $(\mathcal{N}, \rho, \mu_0)$, the P-language it generates is deterministic if for any of its strings w, there is a unique transition sequence σ enabled by μ_0 that generates w: $\rho(\sigma) = w$.) In the setting of [35], partial controllability and full observability are assumed.

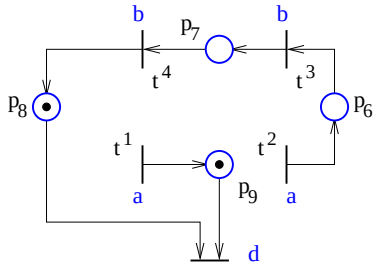


Fig. 5.

5 Disjunctive Constraints

Here we show that under certain boundedness assumptions, disjunctions of constraints can be expressed by conjunctions of constraints by adding not only places, but also transitions to the PN. A disjunction of constraints has the form:

$$\bigvee_i L_i\mu \leq b_i \tag{19}$$

where $L_i \in \mathbb{Z}^{m_i \times n}$ and $b_i \in \mathbb{Z}_i^m$. This can be written as

$$\bigwedge_j \bigvee_{i \in A_j} l_i\mu \leq c_i \tag{20}$$

where $l_i \in \mathbb{Z}^{1\times n}$, $c_i \in \mathbb{Z}$ and A_j is a set of integers. The main idea of our approach is to include additional binary variables δ_i for each constraint $l_i\mu \leq c_i$ such that:

$$[l_i\mu \leq c_i] \leftrightarrow [\delta_i = 1] \tag{21}$$

Then, the disjunction (19) can be replaced by

$$\sum_{i \in A_j} \delta_i \geq 1 \tag{22}$$

for all indices j. If we know that $l_i\mu$ is between the bounds m_i and M_i, (21) is equivalent to the following system of inequalities:

$$l_i\mu + (M_i - c_i)\delta_i \leq M_i \tag{23}$$

$$l_i\mu + (c_i + 1 - m_i)\delta_i \geq c_i + 1 \tag{24}$$

Note that this technique of adding auxiliary variables has been used to solve propositional logic via integer programming in [36, 37]. This technique has also been applied to Hybrid Systems in [38]. In our Petri net context, the variables δ_i will be interpreted as markings of additional "observer" places.

Note also the assumptions that were made:

1. $l_i\mu$ is bounded for all i and all plant markings μ that are reachable (in the closed-loop).
2. some lower bound m_i and upper bound M_i are known for all $l_i\mu$.

The first assumption is reasonable for the specifications that can be implemented in practice. Further, the second assumption appears to be satisfied often in practice.

The algorithm constructing a supervisor is as follows. For each constraint $l_i\mu \leq c_i$ that appears in the disjunction (20), the following operations are done:

1. Let $T_i^+ = \{t : l_i D(\cdot, t) < 0\}$ and $T_i^- = \{t : l_i D(\cdot, t) > 0\}$.
2. Add an additional place d_i and $|T_i^+| + |T_i^-|$ new transitions, as follows. For each transition $t \in T_i^+$, a copy t^+ is added to the PN. (The fact that t^+ is a copy of t means that t^+ satisfies $D^-(\cdot, t^+) = D^-(\cdot, t)$ and $D^+(\cdot, t^+) = D^-(\cdot, t)$.) Further, for each transition $t \in T_i^-$, a copy t^- is added to the PN.
3. For all transitions t^- and t^+, add the arcs (d_i, t^-) and (t^+, d_i) with weight 1.
4. Add the monitor places enforcing (23–24), where δ_i denotes the marking of d_i. Let a_i be the monitor of (23) and e_i the monitor of (24).

Thus, the algorithm enhances the PN with the places d_i, a_i, e_i, and the transitions t^- and t^+. The role of the additional transitions t^- and t^+ is to reset (set) δ_i whenever there is a transition from (to) a marking satisfying $l_i\mu \leq c_i$ to (from) μ' with $l_i\mu' \not\leq c_i$. Note that enforcing the disjunction (20) on the original PN corresponds to enforcing the inequalities (22) on the enhanced PN. Moreover, the enhanced PN can be seen as the composition of a supervisor with the original PN, where the supervisor is a labeled PN. Finally, note that the construction of this algorithm is valid if each $l_i\mu \leq c_i$ is consistent with the observability constraints of the PN. Recall, a sufficient condition that guarantees the observability constraints are satisfied is (8) for free-labeled PNs and (11) for labeled PNs.

The algorithm is illustrated on the following example. Assume we desire to enforce

$$[\mu_2 \leq 0] \vee [\mu_4 \leq 0] \tag{25}$$

on the Petri net of Figure 6(a). Assume also the following bounds are known: $\mu_2 \leq 2$ and $\mu_4 \leq 3$. Note that (25) cannot be represented by conjunctions of inequalities that use only the variables μ_2 and μ_4. For $\mu_2 \leq 2$, the relations (23–24) become (for $c_i = 0$, $m_i = 0$ and $M_i = 2$):

$$\mu_2 + 2\delta_1 \leq 2 \tag{26}$$

$$\mu_2 + \delta_1 \geq 1 \tag{27}$$

Similarly, for $\mu_4 \leq 3$ we have

$$\mu_4 + 3\delta_2 \leq 3 \tag{28}$$

$$\mu_4 + \delta_2 \geq 1 \tag{29}$$

The places d_1 and d_2 are shown in Figure 6(b). Figure 6(c) shows also the monitors a_1, e_1, a_2 and e_2, which correspond to (26–29), in this order. Finally, our disjunction (25) can be implemented by enforcing $\delta_1 + \delta_2 \geq 1$ (Figure 6(d)), if $\delta_1 + \delta_2 \geq 1$ is admissible.

In general, (22) may not be admissible. However, one may attempt transforming it to an admissible form by means of one of the techniques of section 2, such as the approach of [4]. Note also that in our example, the supervisor can be represented as in Figure 7, where the PN of Figure 6(d) can be seen as the composition of the plant in Figure 6(a) and the supervisor. In Figure 7, α_i denotes the label of t_i, for $i = 2, 3, 4, 5$.

6 Decentralized and Distributed Supervision

The decentralized control of PNs is approached in [39, 14]. The setting is as follows. A PN $\mathcal{N} = (P, T, D^-, D^+)$ is given, representing the plant. The plant has m subsystems, each having a set of controllable transitions $T_{c,i} \subseteq T$ and a set of observable transitions $T_{o,i} \subseteq T$, for $i = 1 \ldots m$. In this setting, we are to design m supervisors $\mathcal{S}_i$, each allowed to disable transitions $t \in T_{c,i}$ and observe transitions $t \in T_{o,i}$, such that the joint operation of the supervisors $\mathcal{S}_i$ ensures the specification (1) is satisfied. In [39] a decentralized admissibility concept is introduced, called **d-admissibility**. An efficient structural test for d-admissibility based on (7–8) is given in [39]. Several cases have been studied:

1. The specification (1) is d-admissible.
2. The specification (1) is not d-admissible and communication of transition firings is allowed.
3. The specification (1) is not d-admissible and communication is not allowed or is restricted.

Case 1 is solved by a construction similar to that of (2) and (3). Case 2 is reduced to case 1 by allowing event communication add more elements to the sets $T_{c,i}$ and $T_{o.i}$. However, case 3 is more involved. Assuming no communication is allowed, the problem is to decompose the specification (1) into sets of constraints $L_1\mu \leq b_1 \ \ldots \ L_r\mu \leq b_r$ such that each $L_i\mu \leq b_i$ is d-admissible and

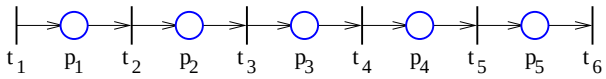
$$(L_1\mu \leq b_1 \wedge L_2\mu \leq b_2 \wedge \ldots L_r\mu \leq b_r) \Rightarrow L\mu \leq b \tag{30}$$

The d-admissibility requirement can be tested by inequalities similar to (7) and (8). In [14] this problem is approached using the parameterization (12–13), by replacing (30) with the conservative requirement that

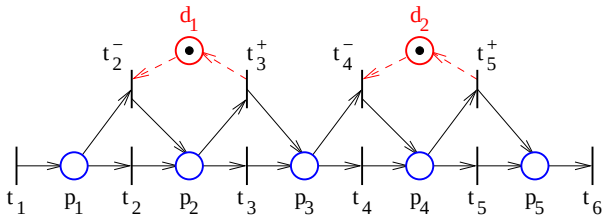
$$L_1 + L_2 + \ldots L_m = R_1 + R_2 L \tag{31}$$

$$b_1 + b_2 + \ldots b_m = R_2(b + \mathbf{1}) - \mathbf{1} \tag{32}$$

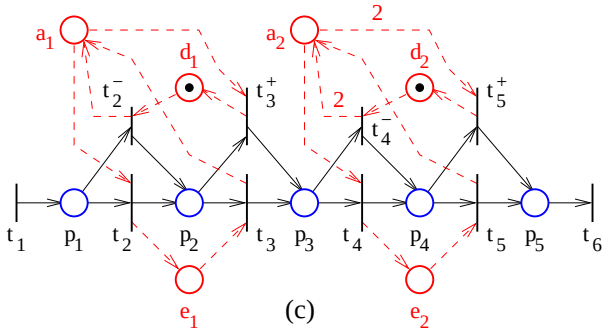
where R_1 has nonnegative integer elements and R_2 is diagonal with positive integer elements on the diagonal. Integer programming is then used to find L_i, b_i, R_1 and R_2. The problem is solved in a similar way when communication is allowed.


t1 p1 t2 p2 t3 p3 t4 p4 t5 p5 t6

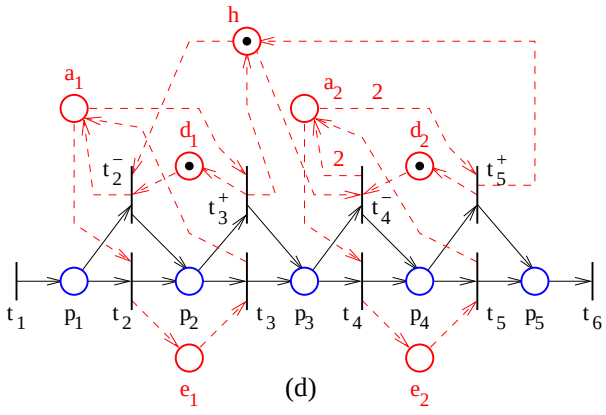
(a)


d1
d2
t2−
t3+
t4−
t5+
t1 p1 t2 p2 t3 p3 t4 p4 t5 p5 t6

(b)


a1
a2
2
d1
d2
t2−
t3+
2
t4−
t5+
t1 p1 t2 p2 t3 p3 t4 p4 t5 p5 t6
e1
e2

(c)


h
a1
a2
2
d1
d2
t2−
2
t5+
t3+
t4−
t1 p1 t2 p2 t3 p3 t4 p4 t5 p5 t6
e1
e2

(d)

Fig. 6.

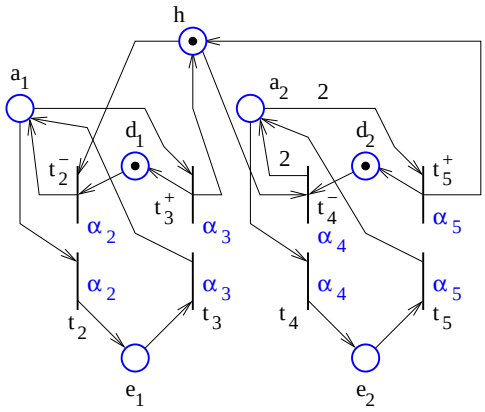


Fig. 7.

7 Conclusion

Based on this survey we may conclude that results are available for a wide variety of supervision problems. Moreover, structural approaches could be used to approach complex supervision problems. By avoiding reachability-graph analysis, structural methods promise computational benefits. On the other hand, for bounded PN plants, reachability analysis could solve any of the specifications considered in this paper. For many of the approaches surveyed here, one of the main limitations is that the liveness problem is ignored. This means that the closed-loop system might reach deadlock/livelock states. While numerous liveness enforcement results exist, the problem is difficult in the general case. Some of the methods presented here have been implemented in software [40]. For the future, we could envision a software tool that analyzes the supervision problem first, and then selects the "best" method that applies.

References

1. Giua, A., DiCesare, F., Silva, M.: Generalized mutual exclusion constraints on nets with uncontrollable transitions. In: Proceedings of the IEEE International Conference on Systems, Man and Cybernetics. (1992) 974–979
2. Yamalidou, E., Kantor, J.: Modeling and optimal control of discrete-event chemical processes using Petri nets. Computers and Chemical Engineering **15** (1991) 503–519
3. Krogh, B., Holloway, L.: Synthesis of feedback control logic for manufacturing systems. Automatica **27** (1991) 641–651
4. Moody, J.O., Antsaklis, P.J.: Supervisory Control of Discrete Event Systems Using Petri Nets. Kluwer Academic Publishers (1998)
5. Tittus, M., Egardt, B.: Hierarchical supervisory control for batch processes. IEEE Transactions on Control Systems Technology **7** (1999) 542–554

6. Giua, A., Seatzu, C.: Supervisory control of railway networks with Petri nets. In: Proceedings of the 40'th IEEE Conference on Decision and Control. (2001) 5004–5009
7. Iordache, M., Antsaklis, P.: Synthesis of supervisors enforcing general linear vector constraints in Petri nets. IEEE Transactions on Automatic Control **48** (2003) 2036–2039
8. Peterson, J.L.: Petri Net Theory and the Modeling of Systems. Englewood Cliffs, New Jersey: Prentice Hall, Inc. (1981)
9. Iordache, M.V., Antsaklis, P.J.: Supervision based on place invariants: A survey. Technical report isis-2004-003, University of Notre Dame (2004) Sumbitted for journal publication.
10. Yamalidou, E., Moody, J.O., Antsaklis, P.J., Lemmon, M.D.: Feedback control of Petri nets based on place invariants. Automatica **32** (1996) 15–28
11. Iordache, M.V., Moody, J.O., Antsaklis, P.J.: Synthesis of deadlock prevention supervisors using Petri nets. IEEE Transactions on Robotics and Automation **18** (2002) 59–68
12. Iordache, M., Antsaklis, P.: Design of T-liveness enforcing supervisors in Petri nets. IEEE Transactions on Automatic Control **48** (2003) 1962–1974
13. Moody, J.O., Antsaklis, P.J.: Petri net supervisors for DES with uncontrollable and unobservable transitions. IEEE Transactions on Automatic Control **45** (2000) 462–476
14. Iordache, M.V., Antsaklis, P.J.: Decentralized control of Petri nets with constraint transformations. In: Proceedings of the 2003 American Control Conference. (2003) 314–319
15. Basile, F., Chiacchio, P., Giua, A.: Supervisory control of Petri nets based on suboptimal monitors places. In: Proceedings of the 4th International Workshop on Discrete Event Systems. (1998) 85–87
16. Basile, F., Chiacchio, P., Giua, A.: On the choice of suboptimal monitor places for supervisory control of Petri nets. In: Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics. (1998) 752–757
17. Basile, F., Chiacchio, P., Giua, A.: Optimal control of Petri net monitors with control and observation costs. In: Proceedings of the 39'th IEEE International Conference on Decision and Control. (2000) 424–429
18. Stremersch, G.: Supervision of Petri Nets. Kluwer Academic Publishers (2001)
19. Chen, H.: Control synthesis of Petri nets based on s-decreases. Discrete Event Dynamic Systems: Theory and Applications **10** (2000) 233–250
20. Chen, H., Hu, B.: Monitor-based control of a class of controlled Petri nets. In: Proceedings of the 3rd International Conference on Automation, Robotics and Computer Vision. (1994)
21. Chen, H.: Net structure and control logic synthesis of controlled Petri nets. IEEE Transactions on Automatic Control **43** (1998) 1446–1450
22. Holloway, L., Krogh, B.: Synthesis of feedback control logic for a class of controlled Petri nets. IEEE Transactions on Automatic Control **35** (1990) 514–523
23. Ramadge, P.: Some tractable supervisory control problems for discrete-event systems modeled by Büchi automata. IEEE Transactions on Automatic Control **34** (1989) 10–19
24. Golaszewski, C.H., Ramadge, P.J.: Mutual exclusion problems for discrete event systems with shared events. In: Proceedings of the 27th IEEE Conference on Decision and Control. (1988) 234–239
25. Li, Y., Wonham, W.: Control of Vector Discrete-Event Systems II - Controller Synthesis. IEEE Transactions on Automatic Control **39** (1994) 512–530

26. Ghaffari, A., Rezg, N., Xie, X.: Feedback control logic for forbidden-state problems of marked graphs: application to a real manufacturing system. IEEE Transactions on Automatic Control **48** (2003) 2–17
27. Ghaffari, A., Rezg, N., Xie, X.: Design of a live and maximally permissive petri net controller using the theory of regions. IEEE Transactions on Robotics and Automation **19** (2003) 137–142
28. Darondeau, P., Xie, X.: Linear control of live marked graphs. Automatica **39** (2003) 429–440
29. Holloway, L., Krogh, B.: On closed-loop liveness of discrete-event systems under maximally permissive control. IEEE Transactions on Automatic Control **37** (1992) 692–697
30. Boel, R.K., Ben-Naoum, L., Breusegem, V.V.: On forbidden state problems for a class of controlled Petri nets. IEEE Transactions on Automatic Control **40** (1995) 1717–1731
31. Holloway, L., Krogh, B.: A generalization of state avoidance policies for controlled Petri nets. IEEE Transactions on Automatic Control **41** (1996) 804–816
32. Zhang, L., Holloway, L.E.: Forbidden state avoidance in controlled Petri nets under partial observation. In: Proceedings of the 33rd Annual Allerton Conference on Communications, Control, and Computing. (1995) 146–155
33. Iordache, M.V.: Methods for the Supervisory Control of Concurrent Systems Based on Petri Net Abstractions. PhD thesis, University of Notre Dame (2003)
34. Giua, A., DiCesare, F.: Blocking and controllability of Petri nets in supervisory control. IEEE Transactions on Automatic Control **39** (1994) 818–823
35. Kumar, R., Holloway, L.: Supervisory control of Petri nets with regular specification languages. IEEE Transactions on Automatic Control **41** (1996) 245–249
36. Williams, H.P.: Linear and integer programming applied to the propositional calculus. International Journal of Systems Research and Information Science **2** (1987) 81–100
37. Williams, H.P.: Model building in mathematical programming. Wiley (1993) (3rd ed.).
38. Bemporad, A., Morari, M.: Control of systems integrating logic, dynamics, and constraints. Automatica **35** (1999) 407–427
39. Iordache, M.V., Antsaklis, P.J.: Admissible decentralized control of Petri nets. In: Proceedings of the 2003 American Control Conference. (2003) 332–337
40. Iordache, M.V., Antsaklis, P.J.: Software tools for the supervisory control of Petri nets based on place invariants. Technical report isis-2002-003, University of Notre Dame (2002)