2005 American Control Conference
June 8-10, 2005. Portland, OR, USA

A Structural Approach to the Enforcement of Language and Disjunctive Constraints

Marian V. Iordache and Panos J. Antsaklis

Abstract— **This paper approaches the supervision of Petri nets with two new results that extend the area of applicability of a method known as supervision based on place invariants (SBPI). The first result deals with the enforcement of specifications expressed by Petri net languages. The second result deals with the enforcement of disjunctive constraints under certain boundedness assumptions. These are significant extensions of the SBPI, as language and disjunctive constraints are more expressive than the type of constraints considered in the past with the SBPI.**

I. Introduction

Petri nets (PNs) are an important class of discrete event systems, allowing a compact representation of concurrent systems. The literature on the supervision of PNs contains numerous references to the enforcement of specifications consisting of linear inequalities on the PN marking, such as [2], [5], [14], [20]. Such constraints have 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) have been used in various applications, such as in the context of the constrained optimal control of chemical processes [19], the coordination of AGVs [10], manufacturing constraints [13], and mutual exclusion in batch processing [16]. Moreover, by considering also classes of constraints that can be reduced to (1) on transformed PNs, specifically the generalized linear constraints of [9], other applications can be mentioned here as well: supervisory control of railway networks [6] and fairness enforcement, such as bounding the difference between the number of occurrences of two events, in protocols [3] and manufacturing [12].

For *safe*[1] PNs, it is known that any forbidden marking specification can be represented by constraints (1), as shown in [20], [5]. However, in the general case, language and state specifications cannot be expressed by constraints (1). In this paper we show that two types of specifications that are more general than (1) can be reduced to specifications (1) on transformed PNs. We will show this for language specifications on labeled PNs and disjunctions of constraints (1). The results for disjunctions of constraints are obtained under some boundedness assumptions. Note that a labeled PN is a PN in which the transitions are labeled with (not necessarily distinct) events, just as in the automata setting. Further, a disjunction of constraints (1) is described by $L_1\mu \leq b_1 \vee L_2\mu \leq b_2 \vee \ldots L_p\mu \leq b_p$, requiring the marking μ to satisfy at least one of $L_i\mu \leq b_i$, $i = 1 \ldots p$.

The paper is organized as follows. The background material necessary for this paper is introduced in section II, while the results are presented in section III. The results of this paper are new. We have included them also in the technical report [8], which is a survey of the SBPI and the SBPI related results.

II. Background

In the SBPI, the system to be controlled is called **plant**, and is assumed to be given in the form of a PN $\mathcal{N} = (P, T, D^-, D^+)$, 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 also denote by $D = D^+ - D^-$ the incidence matrix. The SBPI provides a supervisor enforcing (1) in the form of a PN $\mathcal{N}_s = (P_s, T, D_s^-, D_s^+)$ with

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

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

where D_s is the incidence matrix of the supervisor, $\mu_{0,s}$ the initial marking of the supervisor, and μ_0 is the initial marking of $\mathcal{N}$. The places of the supervisor are called **monitors**. The supervised system, that is the **closed-loop** system, is a PN $\mathcal{N}_c$ of incidence matrix:

$$D_c = \begin{bmatrix} D \\ -LD \end{bmatrix} \tag{4}$$

As an example, in Figure 6(c) a_1 is the monitor enforcing $\mu(p_2) + 2\mu(d_1) \leq 2$ on the plant of Figure 6(b).

In PNs with uncontrollable and unobservable transitions, a supervisor designed as in (2–3) may not be valid, as it may include monitors preventing firings of plant-enabled uncontrollable transitions and monitors with marking varied by firings of closed-loop enabled unobservable transitions. 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. When inadmissible, the constraints

M. V. Iordache is with the School of Engineering & Engineering Technology, LeTourneau University, Longview, TX 75607, USA MarianIordache@letu.edu

P. J. Antsaklis is with the Department of Electrical Engineering, University of Notre Dame, Notre Dame, IN 46556, USA antsaklis.1@nd.edu

The authors gratefully acknowledge the partial support of the National Science Foundation (NSF CCR01-13131).

[1] A PN is *safe* if all reachable markings are binary vectors (i.e. consisting of 0 and 1 elements)

$L\mu \leq b$ are transformed (if possible) to an admissible form $L_a\mu \leq b_a$ such that

$$L_a\mu \leq b_a \Rightarrow L\mu \leq b \tag{5}$$

Then, the supervisor enforcing $L_a\mu \leq b_a$ is admissible, and enforces $L\mu \leq b$ as well.

A sufficient condition for admissibility is that

$$LD(\cdot, T_{uc}) \leq 0 \tag{6}$$

$$LD(\cdot, T_{uo}) = 0 \tag{7}$$

where T_{uc} and T_{uo} are the sets of uncontrollable and unobservable transitions, respectively. These conditions have been used successfully for SBPI design for PNs with uncontrollable and unobservable transitions [13].

PNs in which every transition corresponds to a distinct event are said to be **free-labeled**. A **labeled PN** is a PN enhanced with a labeling function $\rho : T \to 2^{\Sigma} \cup \{\lambda\}$, where Σ is the set of events, ρ the labeling function, and λ the null event. Following the Ramadge-Wonham setting, Σ can be partitioned into controllable and uncontrollable events, $\Sigma = \Sigma_c \cup \Sigma_{uc}$ and observable and unobservable events $\Sigma = \Sigma_o \cup \Sigma_{uo}$. In this setting, when a transition t fires, an event $e \in \rho(t)$ is generated. If $e \in \Sigma_c$ ($e \in \Sigma_o$), the supervisor is able to disable (observe) this event. Note that t is disabled by the supervisor only when all events $e \in \rho(t)$ are disabled. Compared to free-labeled PNs, here two transitions t_1 and t_2 may produce the same event when fired. A supervisor controls/observes transitions indirectly, by disabling/observing events. *Without loss of generality, we can assume that each transition has a unique label.*[2] Then, the conditions (6) and (7) become:

$$\forall t_1, t_2 \in T,\ \rho(t_1) = \rho(t_2) \Rightarrow LD(\cdot, t_1) = LD(\cdot, t_2) \tag{8}$$

$$\forall t \in T,\ \rho(t) \in \Sigma_{uc} \cup \{\lambda\} \Rightarrow LD(\cdot, t) \leq 0 \tag{9}$$

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

Note that (8–10) can be written compactly as $LA \leq 0$, for some matrix A. This means that the same methods used for finding L_a and b_a subject to (5) and (6) or (5–7) can be applied also here, by replacing (6) with $LA \leq 0$.

Generalized constraints [9] are an extension of the constraints (1). They have the form

$$L\mu + Hq + Cv \leq b \tag{11}$$

where q is the firing vector and v the Parikh vector. They are defined as follows. The **firing vector** $q \in \mathbb{N}^{|T|}$ is used to represent in vector form a transition firing: if t' is the transition that fires, then $q(t') = 1$ and $q(t) = 0\ \forall t \neq t'$. The **Parikh vector** is a transition indexed vector $v \in \mathbb{N}^{|T|}$, such that for all transitions t, $v(t)$ indicates how often t has fired since the initialization of the system. An example is shown in Figure 1. The constraints (11) are interpreted as follows. A supervisor enforcing (11) ensures that: (i) all states (μ, v) satisfy $L\mu + Cv \leq b$; (ii) if q is the firing vector of a transition t, $\mu \xrightarrow{t} \mu'$, and $v' = v + q$, then t may fire only if $L\mu + Hq + Cv \leq b$ and $L\mu' + Cv' \leq b$.

In [9], [7] it is shown that:

1) Supervisors enforcing (11) can be easily designed when all transitions are controllable and observable. When there are uncontrollable and unobersvable transitions, the design problem can be reduced to a problem of enforcing constraints of the form (1) on a transformed PN.
2) Any monitor arbitrarily connected to the places of a Petri net can be described as enforcing a constraint of the form (11), where b corresponds to the initial marking of the monitor.
3) In fact, any PN $(\mathcal{N}, \mu_0)$, $\mathcal{N} = (P, T, D^-, D^+)$, can be described by constraints (11), for $L = 0$, $H = D^-$, $C = D^- - D^+$ and $b = \mu_0$.

From point 3) above, we can see that the specifications (11) correspond to the P-type languages of the free-labeled PNs. Following [15], a language $\mathcal{L}$ is a **P-type** PN language if there is a labeled PN with an initial marking such that $\mathcal{L}$ consists of the words associated with the firing sequences enabled by the initial marking.

III. Results

A. Language Constraints

As mentioned above, the results of [9] identify a class of problems involving language specifications that can be reduced to the design of supervisors enforcing (1). For this class of problems, the plants are free-labeled PNs and the specifications are P-type languages of free-labeled PNs. Further, the supervisor is required to ensure that the closed-loop language is a sublanguage of the given specification. This section shows that we can approach in a similar way more general problems, in which neither the plant nor the specification is restricted to be free-labeled.

As an example, consider the PN and the specification shown in Figure 2. 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 3(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].

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

[2] Indeed, transitions t with labels $e_1, \ldots e_k$ can be replaced by k copies of t, $t_1 \ldots t_k$, where each t_i is labeled with e_i. Further, an unlabeled transition can be labeled with λ.

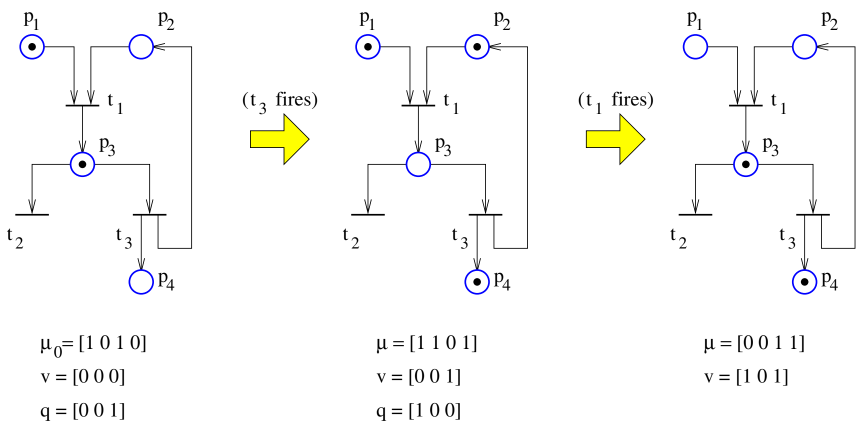

Fig. 1. Illustration of the q and v parameters.

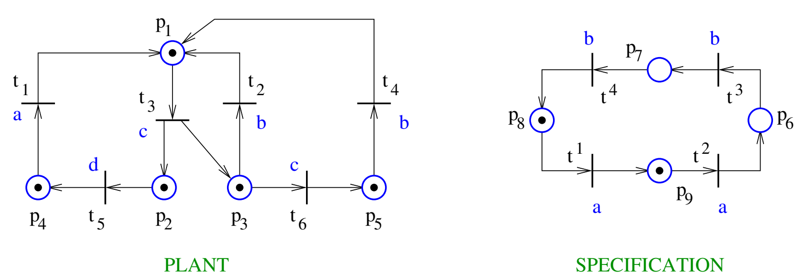

Fig. 2.

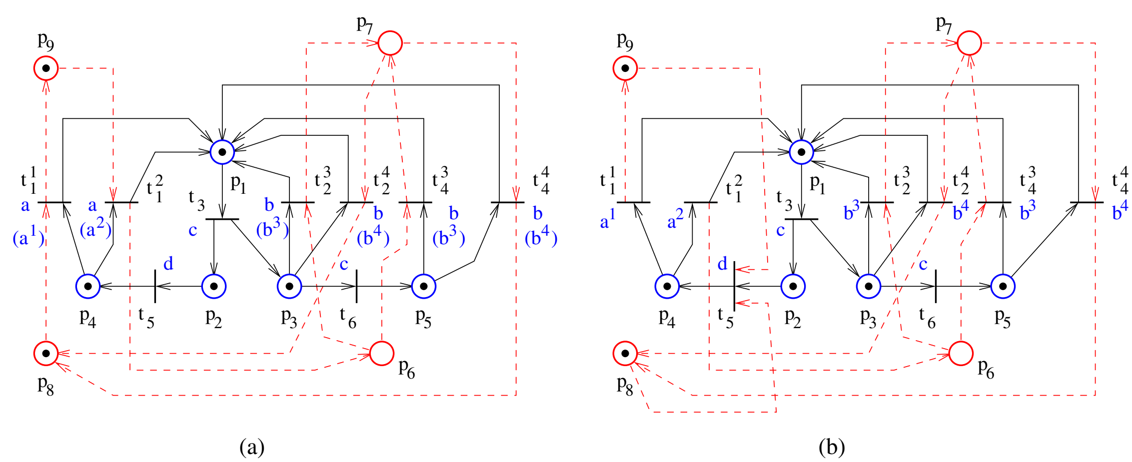

Fig. 3.

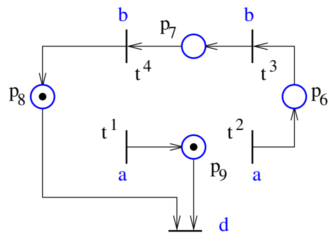

Fig. 4.

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 2, 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 3 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.

As mentioned in the previous section, in the closed-loop, every place of the supervisor corresponds to a specification in terms of constraints (11). 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 available approach for supervisor design in the case of partial controllability and partial observability:

1) Compose the PN plant and the PN specification (supervisor).
2) Relabel the closed-loop, to take in account the supervisor can distinguish between its own transitions.
3) Find the constraints (11) corresponding to the constraints enforced by the monitors of the closed-loop.
4) Transform the constraints (11) to an admissible form, which is at least as restrictive.

We expect to be able to perform step 4) above by adapting the procedure of [9] to account for the admissibility conditions (8–10). These conditions ensure that the connections of a monitor to transitions with the same label are identical.

As an illustration, assume that in our example t_1 (the event a) is uncontrollable but the other transitions are controllable. Assume all other events are observable. Notice that in Figure 3(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 3(b) and Figure 4, respectively. The supervision is admissible, while ensuring the plant generates only words that satisfy the original specification of Figure 2.

Note that in our design the language of the closed-loop is a P-type language, since the supervisor is a labeled PN. However, it is known that the supremal controllable sublanguage of a P-type PN language may not be a P-type PN language [4]. This is an indication that the approach presented here is suboptimal, in the sense that it may not lead to the least restrictive supervisor. 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 [11]. (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 [11], partial controllability and full observability are assumed.

B. Disjunctions of 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:

$$\bigvee_i L_i\mu \leq b_i \tag{12}$$

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

$$\bigwedge_j \bigvee_{i \in A_j} l_i\mu \leq c_i \tag{13}$$

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:

$$[l_i\mu \leq c_i] \leftrightarrow [\delta_i = 1] \tag{14}$$

Then, the disjunction (12) can be replaced by

$$\sum_{i \in A_j} \delta_i \geq 1 \tag{15}$$

for all indices j. If we know that $l_i\mu$ is between the bounds m_i and M_i, (14) is equivalent to the following system of inequalities:

$$l_i\mu + (M_i - c_i)\delta_i \quad \leq \quad M_i \tag{16}$$

$$l_i\mu + (c_i + 1 - m_i)\delta_i \quad \geq \quad c_i + 1 \tag{17}$$

Note that this technique of adding auxiliary variables has been used to solve propositional logic via integer programming in [17], [18]. This technique has also been applied to Hybrid Systems in [1]. 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.

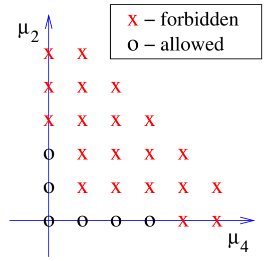

Fig. 5.

The algorithm constructing a supervisor is as follows. For each constraint $l_i \mu \leq c_i$ that appears in the disjunction (13), 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 (16–17), where δ_i denotes the marking of d_i. Let a_i be the monitor of (16) and e_i the monitor of (17).

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 (13) on the original PN corresponds to enforcing the inequalities (15) 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 (7) for free-labeled PNs and (10) for labeled PNs.

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

$$[\mu_2 \leq 0] \vee [\mu_4 \leq 0] \tag{18}$$

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 (18) cannot be represented by conjunctions of inequalities that use only the variables μ_2 and μ_4 (Figure 5). For $\mu_2 \leq 2$, the relations (16–17) become (for $c_i = 0$, $m_i = 0$ and $M_i = 2$):

$$\mu_2 + 2\delta_1 \leq 2 \tag{19}$$
$$\mu_2 + \delta_1 \geq 1 \tag{20}$$

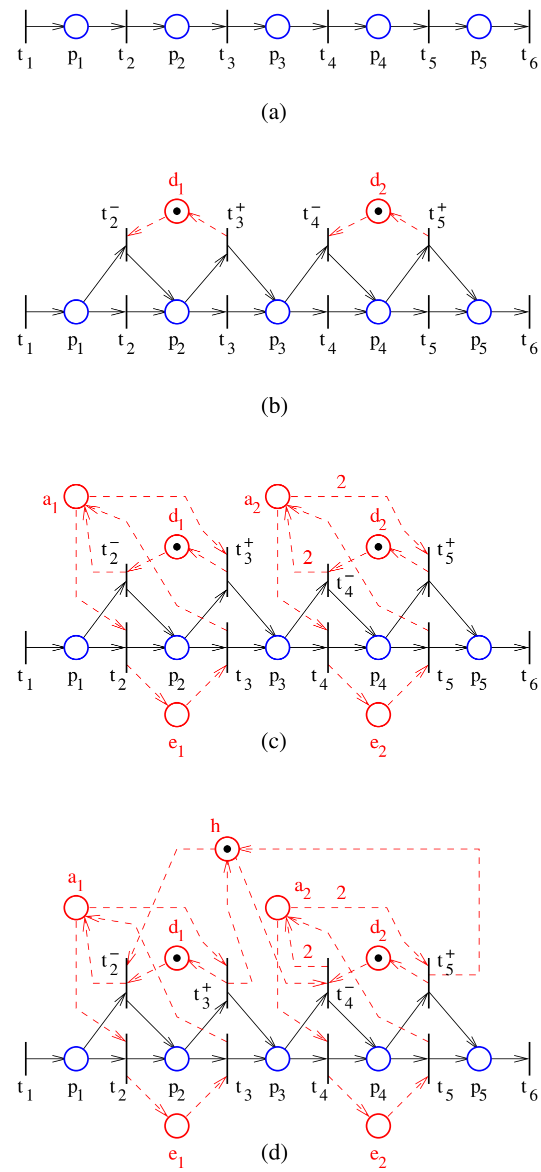

Fig. 6.

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

$$\mu_4 + 3\delta_2 \leq 3 \tag{21}$$
$$\mu_4 + \delta_2 \geq 1 \tag{22}$$

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 (19–22), in this order. Finally, our disjunction (18) 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, (15) may not be admissible. However, one may attempt transforming it to an admissible form by means of the usual SBPI techniques [13]. 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 $i = 2, 3, 4, 5$. The operation enable t_2 in plant if $a_1 \vee (d_1 \wedge h)$

if t_2 fires in plant then
if a_1 /* t_2 fires in closed-loop */
$a_1 \to a_1 - 1$, $e_1 \to e_1 + 1$
else /* t_2^- fires in closed-loop */
$a_1 \to a_1 + 1$, $d_1 \to d_1 - 1$, and $h \to h - 1$
end
end

enable t_3 in plant if $a_1 \vee e_1$

if t_3 fires in plant then
if a_1 /* t_3^+ fires in closed-loop */
$a_1 \to a_1 - 1$, $d_1 \to d_1 + 1$, and $h \to h + 1$
else /* t_3 fires in closed-loop */
$a_1 \to a_1 + 1$, $e_1 \to e_1 - 1$
end
end

Fig. 8. Description of the operation of the supervisor. For simplicity of notation, a_1, e_1, ... denote $\mu(a_1)$, $\mu(e_1)$, ...

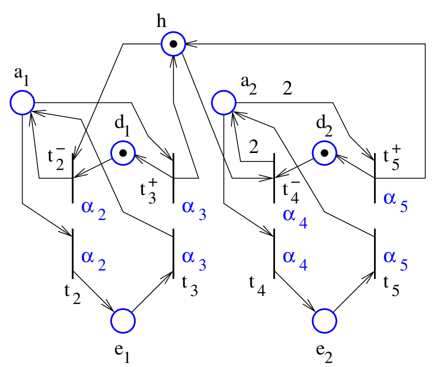

Fig. 7.

of the supervisor can be described as in Figure 8, for the transitions t_2 and t_3. Similar operations are performed for t_4 and t_5.

IV. Conclusions

This paper shows there is an efficient structural solution to problems involving language specifications and disjunctive constraints. The approach of this paper is to reduce these problems to the SBPI problem, which can be solved by various structural methods from the literature. The reduction approach for language specifications appears to be suboptimal. The reduction approach for disjunctive constraints is optimal, but it may result in SBPI problems involving PNs with large weights on the transition arcs. The approach for disjunctive constraints is only possible under certain boundedness assumptions, which are likely to be satisfied in real-world applications.

References

[1] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, 1999.

[2] H. Chen and B. Hu. Monitor-based control of a class of controlled Petri nets. In *Proceedings of the 3rd International Conference on Automation, Robotics and Computer Vision*, 1994.

[3] H. J. Genrich, K. Lautenbach, and P. S. Thiagarajan. Elements of general net theory. In Brauer, W., editor, *Net Theory and Applications*, volume 84 of *Lecture Notes in Computer Science*, pages 21–163. Springer-Verlag, 1980.

[4] A. Giua and F. DiCesare. Blocking and controllability of Petri nets in supervisory control. *IEEE Transactions on Automatic Control*, 39(4):818–823, 1994.

[5] A. Giua, F. DiCesare, and M. Silva. Generalized mutual exclusion constraints on nets with uncontrollable transitions. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, pages 974–979, 1992.

[6] A. Giua and C. Seatzu. Supervisory control of railway networks with Petri nets. In *Proceedings of the 40'th IEEE Conference on Decision and Control*, pages 5004–5009, December 2001.

[7] M. V. Iordache. *Methods for the Supervisory Control of Concurrent Systems Based on Petri Net Abstractions*. PhD thesis, University of Notre Dame, 2003.

[8] M. V. Iordache and P. J. Antsaklis. Supervision based on place invariants: A survey. Technical report isis-2004-003, University of Notre Dame, July 2004.

[9] M.V. Iordache and P.J. Antsaklis. Synthesis of supervisors enforcing general linear vector constraints in Petri nets. *IEEE Transactions on Automatic Control*, 48(11):2036–2039, 2003.

[10] B.H. Krogh and L.E. Holloway. Synthesis of feedback control logic for manufacturing systems. *Automatica*, 27(4):641–651, 1991.

[11] R. Kumar and L. Holloway. Supervisory control of Petri nets with regular specification languages. *IEEE Transactions on Automatic Control*, 41(2):245–249, 1996.

[12] Y. Li and W. Wonham. Control of Vector Discrete-Event Systems I - The Base Model. *IEEE Transactions on Automatic Control*, 38(8):1214–1227, 1993.

[13] J. O. Moody and P. J. Antsaklis. *Supervisory Control of Discrete Event Systems Using Petri Nets*. Kluwer Academic Publishers, 1998.

[14] J. O. Moody and P. J. Antsaklis. Petri net supervisors for DES with uncontrollable and unobservable transitions. *IEEE Transactions on Automatic Control*, 45(3):462–476, 2000.

[15] J. L. Peterson. *Petri Net Theory and the Modeling of Systems*. Englewood Cliffs, New Jersey: Prentice Hall, Inc., 1981.

[16] M. Tittus and B. Egardt. Hierarchical supervisory control for batch processes. *IEEE Transactions on Control Systems Technology*, 7(5):542–554, 1999.

[17] H. P. Williams. Linear and integer programming applied to the propositional calculus. *International Journal of Systems Research and Information Science*, 2:81–100, 1987.

[18] H. P. Williams. *Model building in mathematical programming*. Wiley, 1993. (3rd ed.).

[19] E. Yamalidou and J. Kantor. Modeling and optimal control of discrete-event chemical processes using Petri nets. *Computers and Chemical Engineering*, 15(7):503–519, 1991.

[20] E. Yamalidou, J. O. Moody, P. J. Antsaklis, and M. D. Lemmon. Feedback control of Petri nets based on place invariants. *Automatica*, 32(1):15–28, 1996.