Computation of Admissible Marking Sets in Weighted Synchronization-Free Petri Nets by Dynamic Programming

Ziyue Ma, Guanghui Zhu, Zhiwu Li, Alessandro Giua

September 19, 2019

Abstract

We study the computation of admissible marking sets in generalized Petri nets. We first show that the admissibility checking in generalized Petri net is NP-hard. Then we consider a special subclass of generalized Petri nets called *weighted-synchronization-free nets* in which each transition has at most one input place. For a net in this subclass, we propose a generating function to compute by dynamic programming the set of admissible markings for a given *generalize mutual exclusion constraint*.

Published as:

[Z. Ma, G. Zhu, Z. Li, A. Giua, "Computation of Admissible Marking Sets in Weighted Synchronization-Free Petri Nets by Dynamic Programming", *IEEE Transactions on Automatic Control*, **2020**, 65(6): 2662-2669.]
DOI: 10.1109/TAC.2019.2942570

Ziyue Ma and Guanghui Zhu are with the School of Electro-Mechanical Engineering, Xidian University, Xi'an 710071, China (email: maziyue@gmail.com, zhuguanghui86@gmail.com).

Zhiwu Li is with the School of Electro-Mechanical Engineering, Xidian University, Xi'an 710071, China, and also with the Institute of Systems Engineering, Macau University of Science and Technology, Taipa, Macau (email: zhwli@xidian.edu.cn, systemscontrol@gmail.com).

Alessandro Giua is with the Department of Electrical and Electronic Engineering, University of Cagliari, Cagliari 09124, Italy (email: giua@diee.unica.it).

This work was supported in part by the National Natural Science Foundation of China under Grant Nos. 61472295, 61703321, 61873342, Shaanxi Provincial Natural Science Foundation under Grant No. 2019JQ-022, the Fundamental Research Funds for the Central Universities under Grant JB190407, the Science Technology Development Fund, MSAR, under Grant No. 0012/2019/A3, and the Region Sardinia, FSC 2014 C 2020, project RASSR05871 - MOSIMA.

1 Introduction

Supervisory Control Theory [1] provides a unifying framework for modeling and control of discrete event systems and has been widely applied to model various physical systems. Petri nets have been proposed as SCT models since they can be used to compute an efficient solution to control problems, e.g., *deadlock prevention* [2–5] , *fault diagnosis* [6–8], and *marking avoidance* [9–12] often without requiring the enumeration of the state space. In Petri nets, a state specification consists in a set of *legal markings* and the control objective consists in preventing the system from reaching a *forbidden marking.* A widely-used class of state specifications in Petri nets is the *generalized mutual exclusion constraint* (GMEC) [13] that defines a linear set of legal markings. In a partially controllable Petri net, to enforce a GMEC one needs to find a control policy which enforces a subset of the legal markings, called *admissible markings*, from which the system cannot uncontrollably reach an illegal marking [9].

To solve the aforementioned problem, two types of methods, that we call respectively *reachability-based* and *admissibility-based*, have been proposed. Reachability-based approaches require to enumerate the admissible markings among the reachability set. On the other hand, admissibility-based approaches seek a solution by directly characterizing the set of admissible markings of a given GMEC without enumerating the reachability space. In [14] a method was originally proposed to compute a (possibly strict) subset of admissible markings, and several GMEC transformation algorithms that ensure optimality were later proposed for some restrictive subclasses of *ordinary* Petri nets [15–18]. Although a generalized net can be converted to an equivalent ordinary net, the constraint transformation methods such as [15–17] cannot be applied since in general the converted net does not satisfy some restrictive structural requirements. To the best of our knowledge, there is no method to compute the set of admissible markings in *generalized Petri nets* in the literature. The approach in [14] can be applied to generalized nets but only determines a suboptimal solution, which restricts the evolution of the plant to a subset of the admissible set.

This paper aims to present a methodology for computing the admissible markings set in generalized Petri nets containing uncontrollable transitions. We first show that in generalized Petri nets the problem of determining marking admissibility is NP-hard. Hence we consider a special subclass of generalized Petri nets called *weighted-synchronization-free nets* (WSF nets). The class of WSF nets is a relatively simple subclass of generalized Petri nets of practical interests, which can model systems characterized by operations executed in batches. We propose a *dynamic programming*[1] [19] method to compute the admissible marking set of a given GMEC with nonnegative coefficients if these assumptions are satisfied. Some preliminary results in this setting have been presented in [20], where the GMEC to be enforced is assumed to be elementary and whose influencing subnet is assumed to be a *weighted-state-machine* which is a subclass of WSF nets. In this paper, thanks to the notion of *implicit places*, the two assumptions are partially relaxed and we generalize the approach to a larger class of GMECs and nets.

The paper is organized in seven sections. Section II recalls the basic notions of Petri nets.

[1]Dynamic programming is a recursive procedure that breaks a complex problem into a collection of simpler subproblems. By exploiting the relationship among those subproblems and properly memorizing their solutions, each subproblem is only solved once during the recursion, thus practically achieving efficiency.

Section III studies some properties of admissible marking sets in generalized Petri nets. Section IV introduces weighted synchronization-free nets and elementary GMECs. Section V proposes a dynamic programming method to compute the admissible marking set in a given WSF net. In Section VI an example is presented, while Section VII draws the conclusions.

2 Preliminaries

2.1 Petri Net

A Petri net is a four-tuple $N = (P, T, Pre, Post)$, where P is a set of m *places* represented by circles; T is a set of n *transitions* represented by bars; $Pre : P \times T \to \mathbb{N}$ and $Post : P \times T \to \mathbb{N}$ are the *pre-* and *post-incidence functions* that specify the arcs in the net and are represented as matrices in $\mathbb{N}^{m\times n}$, where $\mathbb{N} = \{0, 1, 2, \ldots\}$. The *incidence matrix* of a net is defined by $C = Post - Pre \in \mathbb{Z}^{m\times n}$, where $\mathbb{Z} = \{0, \pm1, \pm2, \ldots\}$. A Petri net N is said to be *ordinary* if all arcs are unitary, otherwise N is a *generalized* net.

For a transition $t \in T$ we define its *set of input places* as ${}^\bullet t = \{p \in P \mid Pre(p, t) > 0\}$ and its *set of output places* as $t^\bullet = \{p \in P \mid Post(p, t) > 0\}$. The set ${}^\bullet p$ and $p^\bullet$ are analogously defined.

A *marking* is a vector $M : P \to \mathbb{N}$ that assigns to each place of a Petri net a non-negative integer number of tokens. We denote by $M(p)$ the marking of place p. A *marked net* $\langle N, M_0\rangle$ is a net N with an initial marking M_0.

A transition t is *enabled* at M if $M \geq Pre(\cdot, t)$ and may fire reaching a new marking $M' = M_0 + C(\cdot, t)$. We write $M[\sigma\rangle$ to denote that the sequence of transitions σ is enabled at M, and we write $M[\sigma\rangle M'$ to denote that the firing of σ yields M'. We denote by $R(N, M_0)$ the set of all markings reachable from the initial one. We also use $x_1p_1 + \cdots + x_np_n$ to denote the marking $(x_1, \ldots, x_n)^T$ for simplicity.

Given a firing sequence $\sigma \in T^*$, the vector $\mathbf{y}_\sigma$ is the Parikh vector of σ, i.e., $\mathbf{y}_\sigma(t) = k$ if transition t occurs k times in σ.

Node $v_A \in P \cup T$ is said to be in the *upstream* of node $v_B \in P \cup T$ if there exists a sequence $v_Av_1v_2 \cdots v_kv_B$ where $v_i \in P \cup T$ such that $v_A \in {}^\bullet v_1$, $v_i \in {}^\bullet v_{i+1}$ for $i \in \{1, \ldots, k-1\}$ and $v_k \in {}^\bullet v_B$. In such a case node v_B is said to be in the the *downstream* of node v_A.

2.2 GMECs and Partially Controllable PNs

A *Generalized Mutual Exclusion Constraint* (GMEC) is a pair $(\mathbf{w}, k)$ where $\mathbf{w} \in \mathbb{Z}^m$ and $k \in \mathbb{Z}$. A GMEC defines a set of *legal markings*:

$$\mathcal{L}_{(\mathbf{w},k)} = \{M \in \mathbb{N}^m \mid \mathbf{w}^T \cdot M \leq k\}.$$

In this paper it is assumed that the set of legal markings of the plant net is defined by a GMEC $(\mathbf{w}, k)$.

In a partially controllable Petri net, the set of transitions T can be partitioned into a *set of controllable transitions* T_c and a *set of uncontrollable transitions* T_{uc}, i.e., $T = T_c \cup T_{uc}$ and $T_c \cap T_{uc} = \emptyset$. The objective of a *supervisor* is to ensure that only legal markings are reached by preventing transition firings that yield *forbidden markings* in the set $\mathcal{F} = \mathbb{N}^m \setminus \mathcal{L}_{(\mathbf{w},k)}$. However, if net N contains uncontrollable transitions T_{uc}, the supervisor needs to restrict the evolution of the system within the set of *admissible markings* denoted by $\mathcal{A}_{(\mathbf{w},k)}$:

$$\mathcal{A}_{(\mathbf{w},k)} = \{M \in \mathbb{N}^m \mid \forall \sigma_{uc} \in T_{uc}^*, M[\sigma_{uc}\rangle M' \in \mathcal{L}_{(\mathbf{w},k)}\}.$$

In general we have $\mathcal{A}_{(\mathbf{w},k)} \subseteq \mathcal{L}_{(\mathbf{w},k)}$.

The *uncontrollable subnet* of net $N = (P, T, Pre, Post)$ where $T = T_c \cup T_{uc}$ is the net $N' = (P, T_c, Pre', Post')$ obtained from N removing all controllable transitions and their input and output arcs.

In this paper, we assume that all *source transitions*, i.e., transitions t with $|{}^\bullet t| = 0$, are controllable. Since an uncontrollable source transition can generate infinite number of tokens in its output places, such source uncontrollable transitions and their output places can be removed from the net without affecting the control result.

3 Sets of Admissible Markings in Generalized Petri Nets

The sets of admissible markings play an important role in the supervisory control of DES using Petri nets. A supervisor is *maximally permissive* if it disables only the firing of transitions that yield a non-admissible marking, minimally restricting the legal behavior of the plant.

Definition 1 *Given a Petri net $N = (P, T, Pre, Post)$ where $T = T_c \cup T_{uc}$, a marking M, and a GMEC $(\mathbf{w}, k)$, a supervisor is* maximally permissive *if $\forall M \in \mathbb{N}^m$, it holds:*

$$((\forall t \in T_c) M[t\rangle M' \in \mathcal{A}_{(\mathbf{w},k)}) \Leftrightarrow (t \in Ctrl(M))$$

where $Ctrl(M) \subseteq T_c$ is the set of controllable transitions control-enabled at marking M. □

If all transitions are *observable* and the system satisfies the *no concurrency (NC) assumption* [21], a suitable online feedback control policy (which is also maximally permissive) is stated as follows:

1. Let the current marking $M = M_0$; let $Ctrl = T_c$;

2. For each $t \in T_c$ enabled at M, determine if the firing of t yields a marking M' that is not admissible, i.e., $M[t\rangle M' \notin \mathcal{A}_{(\mathbf{w},k)}$: if so, let $Ctrl = Ctrl \setminus \{t\}$;

3. Execute $Ctrl$;

4. Wait until some transition $t \in Ctrl \cup T_{uc}$ fires, update the current marking M_{new} such that $M[t\rangle M_{new}$; let $M = M_{new}$ and $Ctrl = T_c$, go to Step 2.

The key step of this control method is Step 2, which contributes the most to the online computational effort. By reachability analysis one can determine all admissible markings in the reachability set (if bounded). However, solving a reachability problem is computationally expensive and in most cases cannot be done in real-time. On the other hand, pre-computing the set $\mathcal{A}_{(\mathbf{w},k)}$ during the offline stage has several advantages. First, once $\mathcal{A}_{(\mathbf{w},k)}$ is obtained offline, Step 2 can be verified by testing the membership of a marking. Second, since that $\mathcal{A}_{(\mathbf{w},k)}$ does not depend on the initial

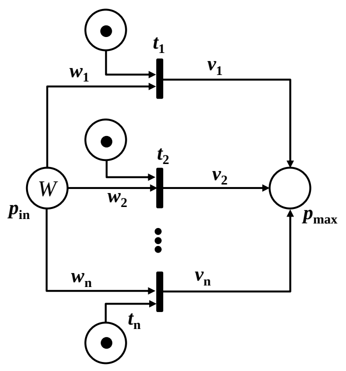

Figure 1: The Petri net constructed in the proof of Theorem 1.

marking of the plant, the control logic does not need to be re-designed when the initial marking changes. Moreover, to compute $\mathcal{A}_{(\mathbf{w},k)}$ only the *influencing subnet* (which will be formally defined in Definition 4 in Section IV) of the plant net needs to be treated, and hence the solution is robust for structural changes of the non-influencing part of the net.

In the following we show that to determine if a marking is admissible in generalized Petri nets is NP-hard by showing that the *Knapsack Problem* [22], which is known to be *NP-complete*, can be reduced to the marking admissibility checking problem in generalized Petri nets.

Problem 1 (Knapsack Problem) *Given a set of n types of unlimited number of items, each type with a weight $w_i \in \mathbb{N}$ and a value $v_i \in \mathbb{N}$, and given a knapsack with a maximal weight limit $W \in \mathbb{N}$ and a total value $V \in \mathbb{N}$, determine if a total value equal to or greater than V can be achieved without exceeding the weight limit W.* □

Theorem 1 *Given a Petri net $N = (P, T, Pre, Post)$ where $T = T_c \cup T_{uc}$, a marking M, and a GMEC $(\mathbf{w}, k)$, to determine if $M \in \mathcal{A}_{(\mathbf{w},k)}$ is NP-hard.*

Proof: Given an Knapsack Problem instance $W, V, w_i, v_i \in \mathbb{N}, i \in \{1, \ldots, n\}$, where n is the number of type of items, we can construct a generalized Petri net N by the procedure described below:

1. Create two places p_{in} and p_{max};
2. Create n transitions $t_1, \ldots, t_n$ satisfying $Pre(p_{in}, t_i) = w_i$, $Post(p_{max}, t_i) = v_i$;
3. Create n places $p_1, \ldots, p_n$ satisfying $Pre(p_i, t_i) = 1$, $Post(p_i, t_i) = 0$;
4. Let $T_{uc} = T = \{t_1, \ldots, t_n\}$;
5. Create a GMEC $(\mathbf{w}, k)$ such that $\mathcal{L}_{(\mathbf{w},k)} = \{M \in \mathbb{N}^m \mid M(p_{max}) \leq V - 1\}$.

It is not difficult to understand that the Knapsack Problem has a positive answer (i.e., there is a combination of items with total weight less or equal to W that has value V or more) if and only if $M = [W, 0, \mathbf{1}]^T \notin \mathcal{A}_{(\mathbf{w},k)}$. This indicates that to determine if $M \in \mathcal{A}_{(\mathbf{w},k)}$ is NP-hard. □

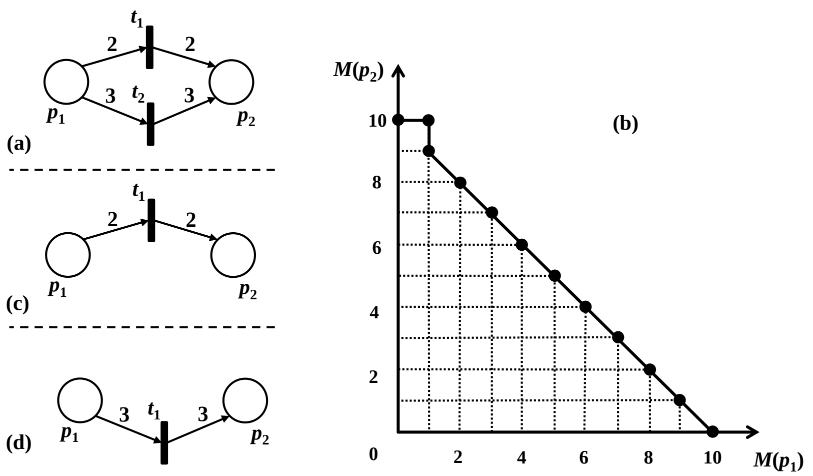

Figure 2: The nets and the illustration of the admissible markings set used in Example 1.

Theorem 1 implies that in generalized Petri nets there does not exist an algorithm with polynomial complexity (if P≠NP) to enforce a given state specification. Moreover, GMEC-based closed-form representation of $\mathcal{A}_{(\mathbf{w},k)}$ can be very complex or may even not exist in some generalized nets. In fact, an OR-AND GMEC that describes the admissible set $\mathcal{A}_{(\mathbf{w},k)}$ of an arbitrary GMEC (even if $\mathbf{w} \geq \mathbf{0}$) can be too complex to be used in reality, which was called the *GMEC inflation* [17]. On the other hand, although in the literature some nonlinear specifications in Petri nets were also proposed [23], they still cannot be used to model an admissible set even in some simple subclasses of generalized Petri nets.

Example 1 *Consider the net in Fig. 2(a) and a GMEC* $(\mathbf{w}, k)$ *defining the legal markings set* $\mathcal{L}_{(\mathbf{w},k)} = \{M \mid M(p_2) \leq 10\}$. *The admissible markings set* $\mathcal{A}_{(\mathbf{w},k)}$, *which is illustrated in Fig. 2(b), can be defined by an OR-AND GMEC:*

$$(M(p_1) + M(p_2) \leq 10) \vee ((M(p_1) \leq 1) \wedge (M(p_2) \leq 10)).$$

However, there is no efficient method (e.g., GMEC transformation) to obtain such a closed-form representation from the initial GMEC $M(p_2) \leq 10$.

On the other hand, one may intuitively think that the marking admissibility can be verified by checking admissibility in the subnets of the uncontrollable subnet. Unfortunately this conjecture is false. Consider the two nets in Figs. 2(c) and 2(d) which are both subnets of the net in Fig. 2(a). In these two cases the admissible markings set can be defined by the so-called stair-GMECs *[23]:*

$$2\lfloor M(p_1)/2 \rfloor + M(p_2) \leq 10 \tag{1}$$

and

$$3\lfloor M(p_1)/3 \rfloor + M(p_2) \leq 10, \tag{2}$$

respectively. However, the marking $(11, 0)$ *satisfies both Eqs.* (1) *and* (2) *but is not admissible.* □

In the rest of paper, we aim to seek for an enumerative approach to pre-compute the admissible marking set in generalized Petri nets. In the next section we propose our strategy and consider a subclass of generalized Petri nets, in which computing the admissible markings is practically feasible.

4 Weighted Synchronization-Free nets and Their Properties

4.1 Weighted Synchronization-Free nets

Definition 2 *A Petri net $N = (P, T, Pre, Post)$ is a* weighted synchronization-free net *(WSF net) if for all $t \in T$, $|^\bullet t| = 1$ holds.* $\square$

WSF nets is more general than the *forward-synchronization-free nets* [15] and *weighted state machines* [20], and can model systems characterized by operations executed in batches, e.g., instead of processing tasks one by one, it waits until a batch of k tasks are available and process them at the same time, and the output of such operation is distributed. Such a primitive is common in many real systems such as transportation, assembly and disassembly systems, etc. Now we introduce the notions of *elementary GMEC* (such specification was called "*element GMEC*" in [24]) and of *influencing subnet.*

Definition 3 *A GMEC $(\mathbf{w}, k)$ is said to be an* elementary GMEC *if there exists a place p_w such that $w(p_w) = 1$ and $w(p) = 0$ for $p \neq p_w$.* $\square$

Since an elementary GMEC restricts the token content of a unique place p_w, in practice we can ignore the places whose tokens will never flow to p_w by firing uncontrollable transitions. This is formalized in the next definition.

Definition 4 *The* influencing subnet *of an elementary GMEC $(\mathbf{w}, k)$ in N, denoted as $N_w = (P_w, T_w, Pre_w, Post_w)$, is the subnet of N obtained by removing all transitions $t \in T_c$ followed by removing all nodes $x \in P \cup T$ such that there does not exist a path from x to p_w. The subnet N_w is also called the influencing subnet of place p_w.* $\square$

Now we can introduce the two assumptions that are used in this paper.

Assumption 1 *The GMEC $(\mathbf{w}, k)$ to be implemented is elementary that restricts the token count of a single place p_w.*

Assumption 1 is a technical assumption that does not restrict the modeling power of the problem too much, since for arbitrary $(\mathbf{w}, k)$ with $\mathbf{w} \geq \mathbf{0}$ we can always introduce a so-called *implicit place* [25] and define a new elementary GMEC $(\mathbf{w}', k)$ on p_w which is equivalent to the original one. Such a procedure is shown in Algorithm 1.

Proposition 1 *Given a Petri $N = (P, T, Pre, Post)$ and a GMEC $(\mathbf{w}, k)$ with initial marking M_0 and $\mathbf{w} \geq \mathbf{0}$, let the net $N' = (P', T', Pre', Post')$ with the initial marking M_0', and the GMEC $(\mathbf{w}', k)$ be the output of Algorithm 1. For any firing sequence σ it holds:*

$$M_0[\sigma\rangle_N \qquad \Longleftrightarrow \qquad M_0'[\sigma\rangle_{N'}$$

Proof: The "$\Longleftarrow$" immediately follows from the fact that (N, M_0) is obtained from (N', M_0') by removing place p_w, and thus all firing sequences in N' are also firable in N. Now we prove "$\Longrightarrow$" by showing that N' simulates N.

Algorithm 1 Problem Refinement

```
Input: A Petri net $N = (P, T, Pre, Post)$ with initial marking $M_0$, a GMEC $(\mathbf{w}, k)$ with $\mathbf{w} \geq \mathbf{0}$
Output: A Petri net $N' = (P', T', Pre', Post')$ with initial marking $M_0'$, and an elementary GMEC $(\mathbf{w}', k)$
1: Let $P' = P \cup \{p_w\}$, let $T' = T$;
2: Let $Pre'(p, t) = Pre(p, t)$ and $Post'(p, t) = Post(p, t)$ for all $(p, t) \in P \times T$;
3: for all $t \in T'$, do
4:     Let $Pre'(p_w, t) = \max\{0, -\mathbf{w}^T \cdot C(\cdot, t)\}$; Let $Post'(p_w, t) = \max\{0, \mathbf{w}^T \cdot C(\cdot, t)\}$;
5: end for
6: Let $M_0'(p_w) = \mathbf{w}^T \cdot M_0$ and $M_0'(p) = M_0(p)$ for $p \neq p_w$;
7: Let $\mathbf{w}'$ be: $w'(p_w) = 1$, $w'(p) = 0$ for $p \neq p_w$;
8: Output $N' = (P', T', Pre', Post')$, $M_0'$, and $(\mathbf{w}', k)$.
```

Assume that $M_0[t\rangle_N M_1$ in the original net, then $M_0 \geq Pre(\cdot, t)$ holds. Since $\mathbf{w} \geq \mathbf{0}$, it also holds:

$$\begin{aligned} M_0'(p_w) &= \mathbf{w}^T \cdot M_0 \\ &\geq \mathbf{w}^T \cdot Pre(\cdot, t) \\ &\geq \mathbf{w}^T \cdot \max\{\mathbf{0}, -C(\cdot, t)\} \\ &= \max\{\mathbf{0}, -\mathbf{w}^T \cdot C(\cdot, t)\} = Pre'(p_w, t), \end{aligned}$$

which means that $M_0'[t\rangle_{N'} M_1'$.

In addition, we observe that $M_1'(p_w) = \mathbf{w}^T \cdot M_0 + C'(p_w, t) = \mathbf{w}^T \cdot M_0 + \mathbf{w}^T \cdot C(\cdot, t) = \mathbf{w}^T \cdot M_1$. By letting markings M_1 and M_1' be two new initial markings, the reasoning above can be repeatedly applied to any firing sequence σ, which concludes the proof. □

By the following theorem we show that the original control problem can be refined to an equivalent problem in which the GMEC to be enforced is elementary.

Theorem 2 *Given a Petri $N = (P, T, Pre, Post)$ and a GMEC $(\mathbf{w}, k)$ with initial marking M_0 and $\mathbf{w} \geq \mathbf{0}$, let the net $N' = (P', T', Pre', Post')$ with the initial marking M_0', and the GMEC $(\mathbf{w}', k)$ be the output of Algorithm 1. For any firing sequence σ it holds:*

$$M_0[\sigma\rangle_N M \in \mathcal{L}_{(\mathbf{w},k)} \quad \Longleftrightarrow \quad M_0'[\sigma\rangle_{N'} M' \in \mathcal{L}_{(\mathbf{w}',k)}$$

Proof: By the proof of Proposition 1, for any firing sequence σ such that $M_0[\sigma\rangle_N M$ and $M_0'[\sigma\rangle_{N'} M'$, $M'(p_w) = \mathbf{w}^T \cdot M$ holds. Since $\mathbf{w}'^T \cdot M' = M'(p_w)$ holds (Step 7 in Algorithm 1), it is clear that:

$$\begin{aligned} M \in \mathcal{L}_{(\mathbf{w},k)} &\Leftrightarrow \mathbf{w}^T \cdot M \leq k \\ &\Leftrightarrow M'(p_w) \leq k \Leftrightarrow M' \in \mathcal{L}_{(\mathbf{w}',k)}. \end{aligned}$$

□

The second assumption concerns the influencing subnet of the elementary GMEC to be enforced.

Assumption 2 *The influencing subnet of the elementary GMEC $(\mathbf{w}, k)$ is WSF and acyclic.*

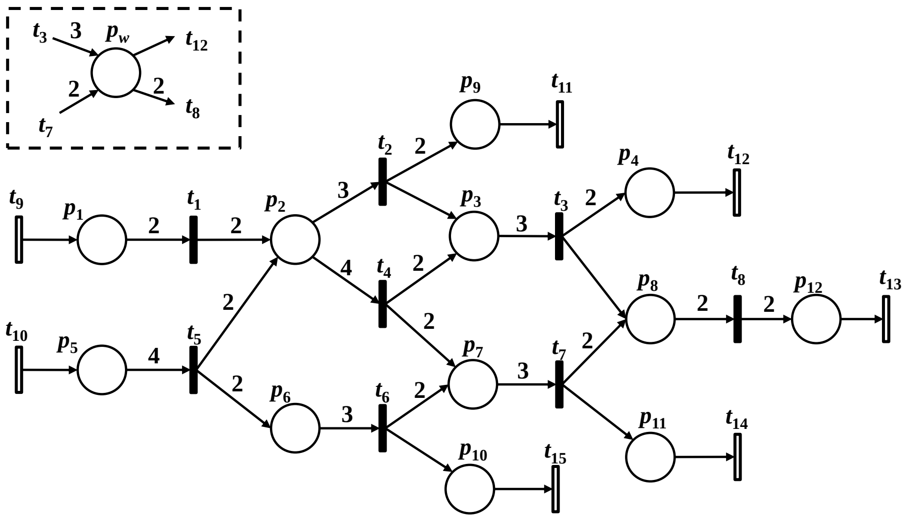

Figure 3: The Petri net used in Example 2.

We believe Assumption 2 is not too restrictive. As we have mentioned in the beginning of this section, many batch-operations in practice such as automatic transportation routine can be modeled by WSF nets. Moreover, these operations are in general loop-free. However, for an arbitrary GMEC, the property of being WSF and acyclic of the original uncontrollable subnet does not guarantee that the net N' and the GMEC $(\mathbf{w}', k)$ obtained by Algorithm 1 satisfy Assumptions 1 and 2. On the other hand, the following proposition provides a way to verify Assumption 2.

Proposition 2 *Given a net N in which the uncontrollable subnet is acyclic and WSF, and given a GMEC $(\mathbf{w}, k)$, the refined control problem obtained by Algorithm 1 satisfies Assumption 2 if and only if for all $t, \hat{t}$ in the uncontrollable subnet, it holds:*

$$\mathbf{w}^T \cdot C(\cdot, t) < 0 \Rightarrow \mathbf{w}^T \cdot C(\cdot, \hat{t}) \leq 0, \forall \hat{t} \textit{ in the dowstream of } t \tag{3}$$

Proof: (Acyclicity) Since the uncontrollable subnet in N is acyclic, the only possibility of introducing cycles by adding place p_w is that there exist two transitions $t, \hat{t}$ in the uncontrollable subnet such that (1) $\hat{t}$ is in the dowstream of t, and (2) there exists an arc from p_w to t (i.e., $w^T \cdot C'(\cdot, t) < 0$) and an arc from $\hat{t}$ to p_w (i.e., $w^T \cdot C'(\cdot, \hat{t}) > 0$) (by Steps 4 to 8 in Algorithm 1), which is excluded by Eq. (3). Hence the influencing subnet of p_w is acyclic.

(WSF property) Since the original uncontrollable subnet in N is WSF, in N' for any transition t that is not synchronization-free, there must exist an arc from p_w to t (i.e., $w^T \cdot C'(\cdot, t) < 0$). Since the uncontrollable subnet in N' is acyclic, we can conclude that t is in the downstream of p_w and hence does not belong to the influencing subnet of p_w. Hence the influencing subnet of p_w is WSF. □

It is worth noticing that if Eq. (3) is not satisfied, the reduction method cannot be applied even if the uncontrollable subnet is a *weighted state machine* (as was the case studied in [20]). Hence the dynamic programming approach which will be proposed in Section V can be applied to any GMEC if (a) it satisfies Eq. (3) in Proposition 2 and (b) the uncontrollable subnet is a WSF net.

Example 2 *The net in Fig. 3 without place p_w models an AGV-transportation system, where each t_i, $1 \leq i \leq 8$, represents an AGV automatically moving parts from its input place to its output places. Hence the set of uncontrollable transitions is $T_{uc} = \{t_1 - t_8\}$. We want to implement a control demand requiring that the number of tokens in p_4 and p_8 should not exceed 2, i.e., the legal marking set is $\mathcal{L}_{(\mathbf{w}_0,k)} = \{M \mid M(p_4) + M(p_8) \leq 2\}$. The uncontrollable subnet in this case consists of places $p_1 - p_{12}$ and transitions $t_1 - t_8$ as well as related arcs.*

Since the uncontrollable subnet satisfy Eq. (3)*, by applying Algorithm 1 one will add a place p_w initially unmarked as shown in the figure. For a new elementary GMEC defining the legal markings $\mathcal{L}_{(\mathbf{w}',k)} = \{M \mid M(p_w) \leq 2\}$, it is not difficult to understand that in the original net the firing of a sequence σ yields a marking that violates $\mathcal{L}_{(\mathbf{w},k)}$ if and only if in the modified net the firing of σ yields a marking that violates $\mathcal{L}_{(\mathbf{w}',k)}$. The influencing subnet of* $(\mathbf{w}', k)$*, which consists of places $p_1, p_2, p_3, p_5, p_6, p_7, p_w$ and transitions $t_1 - t_7$, is acyclic and WSF.* □

4.2 Admissible Bounds in WSF nets

Proposition 3 *Given an elementary GMEC* $(\mathbf{w}, k)$ *bounding place p_w and satisfying Assumption 2, for any place $p \in N_w$ there exists an integer bound $B(p) \in \mathbb{N}$ such that (1) for any marking M such that $M(p) \geq B(p) + 1$, $M \notin \mathcal{A}_{(\mathbf{w},k)}$ holds; (2) $B(p)$ is minimal.*

Proof: Being WSF indicates that each transition has exactly one input place. Thus for any $p \in N_w$, we can always find a sufficiently large integer K such that from $M(p) = K$ we can fire those transitions on the path from p to p_w for sufficient times so that p_w eventually gets more than k tokens, and hence $M \notin \mathcal{A}_{(\mathbf{w},k)}$ when $M(p) = K$. Noticing that the maximal number of tokens that place p_w may receive by firing unobservable transitions is non-decreasing with the increase of the tokens in p, if we increase $M(p)$ from zero one-by-one, we can always reach a minimal integer $B(p)$ satisfying the condition in the statement. □

We call $B(p)$ the *admissible bound* of place p, i.e., any marking M with $M(p) > B(p)$ is not admissible. From Proposition 3 we have the following corollary.

Corollary 1 *Given an elementary GMEC* $(\mathbf{w}, k)$ *bounding place p_w and satisfying Assumption 2, $\mathcal{A}_{(\mathbf{w},k)} \subseteq \mathcal{B}_{(\mathbf{w},k)}$ holds where $\mathcal{B}_{(\mathbf{w},k)} = \{M \in \mathbb{N}^{|P_w|} \mid M(p) \leq B(p)\}$ is the subspace bounded by the admissible bounds.*

By Corollary 1, to compute the set of admissible markings $\mathcal{A}_{(\mathbf{w},k)}$ in an enumerative way it is sufficient to test the markings in the bounded orthogonal integer space $\mathcal{B}_{(\mathbf{w},k)}$, even if the plant net is unbounded (as is the case of the net in Fig. 3). Since the influencing subnet of p_w is acyclic, the admissible bound $B(p_i)$ of each place p_i can be determined by solving the following ILPP based on the state equation:

$$\begin{cases} \min & B(p) = M(p) - 1 \\ s.t. & \mathbf{w}^T \cdot (M + C \cdot \mathbf{y}) \geq k + 1 \\ & M + C \cdot \mathbf{y} \geq \mathbf{0} \\ & M(p') = 0 \text{ for } p' \neq p \\ & \mathbf{y}(i) \geq 0 \text{ for } t_i \in T_{uc} \\ & \mathbf{y}(i) = 0 \text{ for } t_i \in T_c \end{cases} \tag{4}$$

However, testing whether all markings in $\mathcal{B}_{(\mathbf{w},k)}$ are admissible is rather inefficient, because the cardinality of this set, given by $|\mathcal{B}_{(\mathbf{w},k)}| = \prod_{p_i \in N_w}(B(p_i) + 1)$, may be quite large. On the other hand, in the next subsection we propose a dynamic programming method to compute the admissible set, which is proved to be more efficient.

4.3 Generating Function for Dynamic Programming

Consider an elementary GMEC $(\mathbf{w}, k)$ bounding place p_w with influencing subnet $N_w = (P_w, T_w, Pre_w, Post_w)$. We denote a marking $M \in \mathbb{N}^{|P_w|}$ of this subnet as: $M = (\bar{M}, M(p_w))$ where $\bar{M} \in \mathbb{N}^{|\bar{P}_w|}$ is the restriction of M to the places in $\bar{P}_w = P_w \setminus \{p_w\}$.

A function $\mathcal{U} : M \to \mathbb{N}$ maps each marking $M = (\bar{M}, M(p_w)) \in \mathbb{N}^{|P_w|}$ to an integer representing the maximal number of tokens that place p_w may get from M in net N_w. In other words, the maximal number of tokens that place p_w can get in $\langle N_w, M\rangle$ is denoted as $\mathcal{U}(M)$. We extend the domain of $\mathcal{U}$ to arbitrary integer vectors by assuming $\mathcal{U}(M) = undef$ ("not defined") if $M \ngeq \mathbf{0}$ (this cannot happen in a Petri net evolution, but some $M \ngeq \mathbf{0}$ may be encountered in our algorithm during intermediate computation steps). Obviously in an acyclic WSF $\mathcal{U}(\mathbf{0}) = \mathcal{U}(\mathbf{0}, 0) = 0$ holds.

Now we describe the generating function of $\mathcal{U}(\bar{M}, M(p_w))$ which consists of two parts, respectively handling the cases $M(p_w) = 0$ and $M(p_w) > 0$. Here we use $\mathbf{e}_i \in \mathbb{N}^{|\bar{P}_w|}$ to denote a submarking that has a unique token in place p_i, i.e., $\mathbf{e}_i(i) = 1$ and $\mathbf{e}_i(j) = 0$ for $j \neq i$.

Case 1: Markings Satisfying $M(p_w) = 0$

Proposition 4 *Given an influencing subnet N_w that is an acyclic WSF net, for arbitrary marking $M = (\bar{M}, 0)$ the following equation holds:*

$$\begin{aligned} \mathcal{U}(\bar{M}, 0) = \max\{ & \max_{p_i \in \bar{P}_w} \mathcal{U}(\bar{M} - \mathbf{e}_i, 0), \\ & \max_{t \in T_w}(Post(p_w, t) + \mathcal{U}(\bar{M} + \bar{C}(\cdot, t), 0))\}. \end{aligned} \tag{5}$$

where $\bar{C}$ is the incidence matrix of the subnet of N_w by removing place p_w.

Proof: The condition on the right-hand side of Eq. (5) can be partitioned in two parts: (1) $\max_{i \in \{1,\ldots,|P|-1\}} \mathcal{U}(\bar{M} - \mathbf{e}_i, 0)$; (2) $\max_{t \in T_{uc}}(Post(p_w, t) + \mathcal{U}(\bar{M} + \bar{C}(\cdot, t)), 0)$. Now we prove $\geq$ and $\leq$ for Eq. (5).

($\geq$) For part (1), it is not difficult to understand that the maximal number of tokens that p_w may get at $(\bar{M}, 0)$ in the worst case is not less than that at $(\bar{M} - \mathbf{e}_i, 0)$, a marking such that a token is removed from $\bar{M}$.

Now consider part (2). Let $(\bar{M}', Post(p_w, t))$ be a marking such that $(\bar{M}, 0)[t\rangle(\bar{M}', Post(p_w, t))$ where $t \in T_w$. It is obvious that if p_w can get $\mathcal{U}(\bar{M}', 0)$ tokens by firing a sequence σ at $(\bar{M}', 0)$, then p_w can also get $Post(p_w, t) + \mathcal{U}(\bar{M}', 0)$ tokens by firing the sequence $t\sigma$ at $(\bar{M}, 0)$.

($\leq$) By contradiction, suppose that Eq. (5) fails at some $(\bar{M}, 0)$, i.e., place p_w can get r tokens at some marking $(\bar{M}, 0)$ where:

$$r > \max\{ \max_{p_i \in \bar{P}_w} \mathcal{U}(\bar{M} - \mathbf{e}_i, 0), \\ \max_{t \in T_w} (Post(p_w, t) + \mathcal{U}(\bar{M} + \bar{C}(\cdot, t), 0))\}.$$

by firing a sequence $\sigma = t_1 t_2 \cdots t_z$. Now we write such trajectory as

$$(\bar{M}, 0)[t_1\rangle(\bar{M}_1, r_1)[t_2\rangle(\bar{M}_2, r_2) \cdots [t_z\rangle(\bar{M}_z, r_z)$$

where $r_i = \sum_{j=1,\ldots,i} Post(p_w, t_j)$. It is clear that $r_z = \sum_{i=1,\ldots,z} Post(p_w, t_i) = r$ holds.

By part (2) of the condition in Eq. (5) we have $r > Post(p_w, t_1) + \mathcal{U}(\bar{M} + \bar{C}(\cdot, t_1), 0) = Post(p_w, t_1) + \mathcal{U}(\bar{M}_1, 0)$, i.e.:

$$r > \mathcal{U}(\bar{M}_1, 0) + Post(p_w, t_1).$$

Since $\mathcal{U}(\bar{M}_1, 0) \geq Post(p_w, t_2) + \mathcal{U}(\bar{M}_1 + \bar{C}(\cdot, t_2), 0) = Post(p_w, t_2) + \mathcal{U}(\bar{M}_2, 0)$, we have

$$r > \mathcal{U}(\bar{M}_2, 0) + Post(p_w, t_1) + Post(p_w, t_2).$$

This reasoning can be repeatedly applied and finally we have

$$r > \mathcal{U}(\bar{M}_z, 0) + \sum_{i=1,\ldots,z} Post(p_w, t_i),$$

which contradicts the fact that $\sum_{i=1,\ldots,z} Post(p_w, t_i) = r$. □

Case 2: Markings Satisfying $M(p_w) > 0$

Proposition 5 *Given an influencing subnet N_w that is an acyclic WSF net, for arbitrary marking $M = (\bar{M}, M(p_w))$ the following equation holds:*

$$\mathcal{U}(\bar{M}, M(p_w)) = M(p_w) + \mathcal{U}(\bar{M}, 0). \tag{6}$$

Proof: Since the influencing subnet N_w is acyclic, once a token reaches p_w, then it will never be removed from p_w by the firings of any transitions in N_w. Hence all tokens that p_w can receive by firing uncontrollable transitions are those in the upstream places of p_w (i.e., $\mathcal{U}(\bar{M}, 0)$) plus those tokens already in p_w (i.e., $M(p_w)$). □

Example 3 (Ex. 2 cont.) *Consider again the net in Fig. 3 where the influencing subnet of place p_w consists of places $p_1, p_2, p_3, p_5, p_6, p_7, p_w$ and transitions $t_1 - t_7$. Let us first consider a marking $M = (0, 0, 4, 0, 0, 4, 0)$, i.e., $M = 4 \cdot p_3 + 4 \cdot p_7$. On one hand, $\mathcal{U}(M)$, the maximal number of tokens that p_w may get, is necessarily at least the same as $\mathcal{U}(M')$ and $\mathcal{U}(M'')$ where $M' = (0, 0, 3, 0, 0, 4, 0)$ and $M'' = (0, 0, 4, 0, 0, 4, 0)$, since M' and M'' are markings such that a token is removed from M. On the other hand, since the firings of t_3 and t_7 yield marking $M' = (0, 0, 1, 0, 0, 4, 3)$ and marking $M'' = (0, 0, 4, 0, 0, 1, 2)$, respectively, place p_w can get at least $\mathcal{U}(0, 0, 1, 0, 0, 4, 0) + 3$ or $\mathcal{U}(0, 0, 4, 0, 0, 1, 0) + 2$ tokens by firing some sequences from M' and M'', respectively.* □

5 Computation of Admissible Marking Set in WSF Nets by Dynamic Programming

In the following we propose a dynamic programming algorithm to compute the admissible marking set of an elementary GMEC satisfying Assumptions 1 and 2. In the algorithm we use a $|P_w|$-dimensional integer array to store the intermediate result of $\mathcal{U}$. The array is denoted as $\mathcal{U}[x_1, x_2, \ldots, x_n]$ and each of its element $\mathcal{U}[\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n]$ stores the value of $\mathcal{U}(M)$ where $M = (\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n)$. We use a particular integer $NA(NA > k)$ to denote a sufficient large integer: if $\mathcal{U}[M] \neq NA$ then M is admissible. We also use a particular negative integer (e.g., -1) to mark an element that is *unknown*, i.e., it is not computed yet.

Algorithm 2 Computation of the Admissible Marking Set

```
Input: An elementary GMEC $(\mathbf{w}, k)$ bounding place $p_w$ whose influencing subnet $N_w$ is an acyclic
    WSF net
Output: The admissible marking set $\mathcal{A}_{(\mathbf{w},k)}$
 1: Create array $\mathcal{U}[x_1, \ldots, x_{|P_w|}]$ of size $[B(p_1)+2, \ldots, B(p_{|P_w|})+2]$. Initialize all elements as
    unknown;
 2: Let $\mathcal{U}[\mathbf{0}, 0] = 0$;
 3: while unknown in $\mathcal{U}[\bar{M}, 0]$ exists, do
 4:     select an unknown element $\mathcal{U}[\bar{M}, 0]$ such that:
            $\begin{cases} \forall p_i \in P_w \setminus \{p_w\}, \mathcal{U}[\bar{M} - \mathbf{e}_i, 0] \text{ is not } unknown, \\ \forall t \in T_w, \mathcal{U}[\bar{M} + \bar{C}(\cdot, t), 0] \text{ is not } unknown \end{cases}$
 5:     Compute $\mathcal{U}[\bar{M}, 0]$ according to Eq. (5);
 6:     If $\mathcal{U}[\bar{M}, 0] = NA$, then let $\mathcal{U}[\bar{M}', 0] = NA$ for all $M' \geq M$;
 7: end while
 8: for all unknown in $\mathcal{U}[\bar{M}, M(p_w)]$ where $M(p_w) > 0$ do
 9:     let $\mathcal{U}[\bar{M}, M(p_w)] = M(p_w) + \mathcal{U}[\bar{M}, 0]$;
10:     if $\mathcal{U}[\bar{M}, M(p_w)] > k$, let $\mathcal{U}[\bar{M}, M(p_w)] = NA$;
11: end for
12: Output $\mathcal{A}_{(\mathbf{w},k)} = \{M \in \mathcal{U} \mid \mathcal{U}[M] \neq NA\}$ that is the admissible marking set.
```

By Algorithm 2, a $|P_w|$-dimensional array $\mathcal{U}$ is first created where all elements are initialized as *unknown*. Then the algorithm fills the subarray $\mathcal{U}[\bar{M}, 0]$ starting from $[\mathbf{0}, 0]$. In each iteration an *unknown* element is computed according to Eq. (5). We note that the max{} operator ignores any *undef* entry, i.e., *undef* does not participate in the computation while it returns NA if any of its entries has a value NA and/or if any of its entries has a subscript $i > B(p_i)$.

Once all $\mathcal{U}[\bar{M}, 0]$ elements are computed, it continues to fill the full array $\mathcal{U}$ according to Eq. (6). Step 6 ensures that once an element $\mathcal{U}[\bar{M}, 0]$ is determined to be not admissible, then all elements $\mathcal{U}[\bar{M}', 0]$ beyond $\mathcal{U}[\bar{M}, 0]$ are also marked as NA. When all elements of $\mathcal{U}$ are computed, it outputs the set of admissible markings $\mathcal{A}_{(\mathbf{w},k)}$. The following theorem guarantees that Step 4 necessarily terminates in a finite number of steps to find a selectable element.

Theorem 3 *In Step 4 of Algorithm 2, the following procedure always halt in a finite number of steps and outputs a selectable element.*

1. *Choose an arbitrary unknown element* $(M_1, 0)$.
2. *If* $(M_i, 0)$ *is not selectable due to some unknown elements in Eq.* (5)*, arbitrarily choose an unknown element* $(M_{i+1}, 0)$ *from them. This procedure is repeated until a selectable element* $(M_r, 0)$ *is reached.*

Proof: First, let us add $|P_w|$ transitions $t'_1, \ldots, t'_{|P_w|}$ to net N_w such that each transition t'_i removes a single token from place p_i (i.e., $Pre(p_i, t'_i) = 1$), yielding a net N'' that is still acyclic.

Now we prove that the procedure above will eventually terminate. Suppose, by contradiction, that the procedure does not terminate. Since $\mathcal{U}$ is finite, among all *unknown* elements there exists a cyclic sequence:

$$(\bar{M}_1, 0) \rightarrow (\bar{M}_2, 0) \rightarrow \ldots \rightarrow (\bar{M}_x, 0) \rightarrow (\bar{M}_1, 0).$$

This indicates that there exists a T-invariant of the net N'' such that $(\bar{M}_1, 0)[\sigma\rangle(\bar{M}_1, 0)$, which contradicts the fact that N'' is acyclic and does not contain any source transitions. As a result, the procedure in the statement terminates. □

Corollary 2 *Algorithm 2 is correct and terminates in a finite number of steps.*

Proof: The correctness of Algorithm 2 is guaranteed by Propositions 4 and 5. For termination, since Step 4 terminates normally by Theorem 3 and in each iteration at least one *unknown* element is computed, Algorithm 2 halts in at most $\prod_{p_i \in P_w}(B(p_i) + 2))$ iterations. □

Here we briefly discuss the complexity of Algorithm 2. By Corollary 2 the number of elements to be computed is $O(\prod_{p_i \in P_w}(B(p_i))) = O(|\mathcal{B}_{(\mathbf{w},k)}|)$. Although Algorithm 2 computes $\mathcal{U}(M)$ for all admissible markings in an enumerative way, simulation results in Section VI shows that it can be done efficiently in practice, mainly due to the fact that only scalar comparisons and additions are performed.

Remark 1 *Once* $\mathcal{A}_{(\mathbf{w},k)}$ *is obtained from Algorithm 2, a straightforward online control algorithm is to check if the firing of each controllable transition* t *leads to some marking not in* $\mathcal{A}_{(\mathbf{w},k)}$*. However, the online computation load could be high since the size of* $\mathcal{A}_{(\mathbf{w},k)}$ *may be very large. On the other hand, we can collect its maximum element to form a* maximal admissible marking set ${}^{\uparrow}\mathcal{A}_{(\mathbf{w},k)}$*, i.e.:*

$$^{\uparrow}\mathcal{A}_{(\mathbf{w},k)} = \{M \in \mathcal{A}_{(\mathbf{w},k)} \mid \nexists M' \in \mathcal{A}_{(\mathbf{w},k)}, M' \gneq M\}. \tag{7}$$

The computation of ${}^{\uparrow}\mathcal{A}_{(\mathbf{w},k)}$ *can be done by traversing the array* $\mathcal{U}$ *(the explicit presentation of* $\mathcal{A}_{(\mathbf{w},k)}$*) only once while collecting* M *such that*

$$(\mathcal{U}[M] \neq NA) \wedge (\forall i \in \{1, \ldots, |P_w|\}, \mathcal{U}[M + \mathbf{e}_i] = NA),$$

a procedure that can be embedded in Algorithm 2. The control decision at marking M *is: transition* t *is permitted if there exists* $M' \in {}^{\uparrow}\mathcal{A}_{(\mathbf{w},k)}$ *such that* $M[t\rangle M'' \leq M'$. □

Remark 2 *Note that the results on the generating function (Propositions 4 and 5) holds for all generalized nets. Furthermore, under some conditions this method can also be extended to arbitrary generalized Petri nets containing cycles and/or synchronizations (i.e., some transitions may have multiple input places) after some modifications.*

If the influencing subnet contains cycles, Theorem 3 no longer holds since there may exist a cyclic sequence of elements in $\mathcal{U}$ such that none of them can be selected by Step 4 in Algorithm 2. This problem can be overcome by randomly picking a marking among them followed by a reachability analysis in the uncontrollable subnet (which is in general much smaller than the whole net) to break such a cycle.

On the other hand, if the influencing subnet contains synchronizations*, i.e., $|{}^\bullet t| > 1$, the admissible bounds for $\mathcal{B}_{(\mathbf{w},k)}$ do not exist, and hence Algorithm 2 cannot be applied. However, if in the reachability set all places p_i in the influencing subnet are bounded by a finite integer $K(p_i)$, we can use $K(p_i)$ instead of $B(p_i)$ such that the proposed dynamic programming method can still be applied.* $\square$

Remark 3 *Finally, we note that this method can also be extended to the case where multiple GMECs (also called AND-GMECs [10]) are to be enforced. Given l GMECs $(\mathbf{w}_i, k_i)$, $1 \le i \le l$, we can compute the set of admissible markings $\mathcal{A}_{(\mathbf{w}_i,k_i)}$ for each of them by using Algorithm 2. The intersection of these sets, i.e., $\bigcap_{i=1}^{l} \mathcal{A}_{(\mathbf{w}_i,k_i)}$, is the admissible marking set of the multiple GMECs.* $\square$

6 Performance Analysis

Still consider the Petri net in Example 3 in Fig. 3 with place p_w, where the influencing subnet of place p_w consists of places $p_1, p_2, p_3, p_5, p_6, p_7, p_w$ and transitions $t_1 - t_7$. By Eqs. (5) and (6) the generating function is:

$$\begin{cases} \mathcal{U}(\mathbf{0},0) = 0; \\ \mathcal{U}(M,0) = \max \begin{cases} \mathcal{U}(M-\mathbf{e}_1,0), \mathcal{U}(M-\mathbf{e}_2,0), \\ \mathcal{U}(M-\mathbf{e}_3,0), \mathcal{U}(M-\mathbf{e}_5,0), \\ \mathcal{U}(M-\mathbf{e}_6,0), \mathcal{U}(M-\mathbf{e}_7,0), \\ \mathcal{U}(M+C(\cdot,t_1),0), \mathcal{U}(M+C(\cdot,t_2),0) \\ 3+\mathcal{U}(M+C(\cdot,t_3),0), \\ \mathcal{U}(M+C(\cdot,t_4),0), \\ \mathcal{U}(M+C(\cdot,t_5),0), \mathcal{U}(M+C(\cdot,t_6),0) \\ 2+\mathcal{U}(M+C(\cdot,t_7),0) \end{cases} \\ \mathcal{U}(M, M(p_w)) = M(p_w) + \mathcal{U}(M,0). \end{cases} \tag{8}$$

We consider a parameterized legal marking set $\mathcal{L}_{(\mathbf{w},k)} = \{M \mid M(p_w) \le k\}$. The computation for different values of k is carried out on a laptop computer with Intel i5-4200M 2.5 GHz processor and 8 GB DDR3 1600Hz RAM, and the results are summarized in Table 1. With the increase of k

k	$\mathcal{B}_{(\mathbf{w},k)} = [B(p_1), B(p_2), B(p_3), B(p_5), B(p_6), B(p_7), B(p_w)]$	$\lvert\mathcal{B}_{(\mathbf{w},k)}\rvert$	$\lvert\mathcal{A}_{(\mathbf{w},k)}\rvert$	$\lvert{}^{\uparrow}\mathcal{A}_{(\mathbf{w},k)}\rvert$	Time [s]	ILPP Time [s]
2	$[7, 11, 6, 8, 2, 5, 2]$	73920	11840	53	3.5	1389
3	$[7, 11, 7, 8, 5, 5, 3]$	323400	22800	79	9.5	3099
4	$[7, 15, 7, 14, 5, 8, 4]$	1646400	44488	145	53	8988
5	$[11, 15, 11, 14, 5, 8, 5]$	5082000	80592	217	172	18331
6	$[11, 15, 11, 17, 8, 11, 6]$	16291440	134816	325	590	*o.t.*
7	$[11, 19, 11, 17, 8, 11, 7]$	24075128	227568	519	1236	*o.t.*
8	$[11, 19, 11, 23, 8, 14, 8]$	47377792	356464	705	2818	*o.t.*

Table 1: The performance of the dynamic programming method and the method based on ILPP for Example 2.

from 2 to 8, the cardinality of the potential admissible set $\mathcal{B}_{(\mathbf{w},k)}$ grows from 73,290 to 47,377,792. For $k = 8$ with the largest space $\mathcal{B}_{(\mathbf{w},k)}$, our method terminates in 2,818 sec (about 45 min) and output the 705 maximal admissible markings in $\mathcal{A}_{(\mathbf{w},k)}$.

The last column is the time to compute $\mathcal{A}_{(\mathbf{w},k)}$ by testing all markings in $\mathcal{B}_{(\mathbf{w},k)}$ by ILPP 4, using an efficient commercial ILPP solver *GUROBI* (academic license). In the ILPP-based method we also utilize the property that is: if a marking is found inadmissible then no ILPP is solved for all greater markings. It is clear that the dynamic programming method proposed in this work is much more efficient than the ILPP-based one. For instance, for $k = 4$ to analyze $\mathcal{B}_{(\mathbf{w},k)}$ consisting of 5,082,000 markings, our dynamic programming method terminates in 3 minutes while the ILPP-based program takes about 5 hours ($\sim 1 : 100$), producing the same result.

7 Conclusion

In this paper we propose a method based on dynamic programming to efficiently compute the admissible marking set of a given GMEC in generalized Petri nets, whose influencing subnet is acyclic and WSF. The set of admissible markings computed by our approach can be further used for the design of liveness-enforcing supervisors. The problem of designing a maximally permissive supervisor in an arbitrary generalized Petri net remains open: this will be the objective of our further research.

References

[1] P. J. Ramadge and W. M. Wonham, "The control of discrete event systems," *Proceedings of IEEE*, vol. 77, no. 1, pp. 81–98, 1989.

[2] Z. W. Li and M. C. Zhou, "Elementary siphons of Petri nets and their application to deadlock prevention in flexible manufacturing systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, vol. 34, no. 1, pp. 38–51, 2004.

[3] Z. W. Li, N. Q. Wu, and M. C. Zhou, “Deadlock control of automated manufacturing systems based on Petri nets – A literature review,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 42, no. 4, pp. 437–462, 2012.

[4] A. Nazeem and S. Reveliotis, “Maximally permissive deadlock avoidance for resource allocation systems with r/w-locks,” *Discrete Event Dynamic Systems*, vol. 25, no. 1, pp. 31–63, 2015.

[5] S. Reveliotis and Z. Fei, “Robust deadlock avoidance for sequential resource allocation systems with resource outages,” *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 4, pp. 1695–1711, 2017.

[6] M. Cabasino, A. Giua, and C. Seatzu, “Fault detection for discrete event systems using Petri nets with unobservable transitions,” *Automatica*, vol. 46, no. 9, pp. 1531–1539, 2010.

[7] M. P. Cabasino, A. Giua, C. N. Hadjicostis, and C. Seatzu, “Fault model identification and synthesis in Petri nets,” *Discrete Event Dynamic Systems*, vol. 25, no. 13, pp. 419–440, 2015.

[8] F. Basile, M. P. Cabasino, and C. Seatzu, “State estimation and fault diagnosis of labeled time Petri net systems with unobservable transitions,” *IEEE Transactions on Automatic Control*, vol. 60, no. 4, pp. 997–1009, 2015.

[9] M. V. Iordache and P. J. Antsaklis, “Supervision based on place invariants: A survey,” *Discrete Event Dynamic Systems*, vol. 16, no. 4, pp. 4451–4492, 2006.

[10] Z. Y. Ma, Z. W. Li, and A. Giua, “Design of optimal Petri net controllers for disjunctive generalized mutual exclusion constraints,” *IEEE Transactions on Automatic Control*, vol. 60, pp. 1774–1785, 2015.

[11] F. Basile, R. Cordone, and L. Piroddi, “A branch and bound approach for the design of decentralized supervisors in Petri net models,” *Automatica*, vol. 52, pp. 322–333, 2015.

[12] J. L. Luo, H. J. Ni, W. M. Wu, S. G. Wang, and M. C. Zhou, “Simultaneous reduction of Petri nets and linear constraints for efficient supervisor synthesis,” *IEEE Transactions on Automatic Control*, vol. 60, no. 1, pp. 88–103, 2015.

[13] A. Giua, F. DiCesare, and M. Silva, “Generalized mutual exclusion constraints for Petri nets with uncontrollable transitions,” in *Proceedings of the IEEE Int. Conf. on Systems, Man, and Cybernetics*, Chicago, USA, 1992, pp. 947–949.

[14] J. Moody and P. Antsaklis, “Petri net supervisors for DES with uncontrollable and unobservable transitions,” *IEEE Transactions on Automatic Control*, vol. 45, no. 3, pp. 462–476, 2000.

[15] J. L. Luo, W. M. Wu, H. Y. Su, and J. Chu, “Supervisor synthesis for enforcing a class of generalized mutual exclusion constraints on Petri nets,” *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, vol. 39, no. 6, pp. 1237–1246, 2009.

[16] S. G. Wang, C. Y. Wang, and M. C. Zhou, “Design of optimal monitor-based supervisors for a class of Petri nets with uncontrollable transitions,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 43, no. 5, pp. 1248–1255, 2013.

[17] Z. Y. Ma, Z. W. Li, and A. Giua, "Characterization of admissible marking sets in Petri nets with conflicts and synchronizations," *IEEE Transactions on Automatic Control*, vol. 62, no. 3, pp. 1329–1341, 2017.

[18] J. Luo and M. Zhou, "Petri-net controller synthesis for partially controllable and observable discrete event systems," *IEEE Transactions on Automatic Control*, vol. 62, pp. 1301–1313, 2017.

[19] R. Bellman, *Dynamic Programming.* Princeton, NJ, USA: Princeton University Press, 1957.

[20] Z. Ma, Z. Li, and A. Giua, "Computation of admissible marking sets in weighted state machines by dynamic programming," in *Proceedings of the 56th IEEE Conference on Decision and Control*, Melbourn, Australia, 2017, pp. 4847–4852.

[21] L. E. Holloway, B. H. Krogh, and A. Giua, "A survey of Petri net methods for controlled discrete event systems," *Discrete Event Dynamic Systems: Theory and Applications*, vol. 7, no. 2, pp. 151–190, 1997.

[22] S. Martello and P. Toth, *Knapsack Problems: Algorithms and Computer Implementations.* New York, NY, USA: John Wiley & Sons, Inc., 1990.

[23] Z. Y. Ma, Z. W. Li, and A. Giua, "Petri net controllers for generalized mutual exclusion constraints with floor operators," *Automatica*, vol. 74, pp. 238–246, 2016.

[24] J. Luo, F. Jin, and C. Huang, "Combined supervisor synthesis based on constraint transformation," in *Proceedings of the 2007 IEEE International Conference on Integration Technology*, Shenzhen, China, 2007, pp. 87–92.

[25] J. M. Colom and M. Silva, "Improving the linearly based characterization of p/t nets," in *Advances in Petri Nets 1990*, G. Rozenberg, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1991, pp. 113–145.