DECENTRALIZED CONTROLLER DESIGN TO ENFORCE BOUNDEDNESS, LIVENESS, AND REVERSIBILITY IN PETRI NETS

Aydın Aybar and Altuğ İftar

Department of Electrical and Electronics Engineering
Anadolu University
26470 Eskişehir, Turkey
aaybar@anadolu.edu.tr aiftar@anadolu.edu.tr
Fax: +90 222 323 9501

Keywords: Discrete-event systems, Petri nets, decentralized control, supervisory controller design, overlapping decompositions.

Abstract

Decentralized supervisory controller design based on overlapping decompositions to enforce boundedness, liveness, and reversibility is considerered for ordinary Petri nets with weighted arcs. In the proposed approach, the given Petri net is first decomposed into a number of overlapping Petri subnets and then expanded such that each Petri subnet appear as disjoint. A controller for each disjoint Petri subnet is next designed to enforce boundedness, liveness, and reversibility in that Petri subnet. Since each subnet is smaller than the original Petri net, this step is, in general, much easier than obtaining a centralized controller for the original Petri net. These controllers are then combined to obtain a controller for the expanded Petri net. Finally, the controller for the expanded Petri net is contracted to obtain a controller for the original Petri net. It is shown that this final controller enforces boundedness, liveness, and reversibility in the original Petri net.

1 Introduction

Petri nets have become a useful tool to model discrete-event systems (DESs) [14]. Various approaches have been developed in the control literature for controlling DESs modelled by Petri nets (e.g., see [7, 12, 5, 15, 6, 3]). Decentralized control approaches for DESs have been developed in [13, 9, 1, 4], besides others. Among these, [1, 4] use Petri net modeling and overlapping decompositions.

Overlapping decompositions were first introduced for continuous-state systems (systems described by differential or difference equations with continuous state variables) in [10]. It has been demostrated that, overlapping decompositions may be used successfully for decentralized controller design for systems whose subsystems are strongly connected through certain dynamics (*the overlapping part*) but weakly connected otherwise (e.g., see [11, 8]). The overlapping decompositions approach was first considered for DESs (as a special case for hybrid systems) in [9]. This approach was then developped for Petri nets in [2].

In this work, we consider decentralized controller design in ordinary Petri nets with weighted arcs, using overlapping decompositions, to enforce boundedness, liveness, and reversibility. Boundedness, liveness, and reversibility may be important in many DESs which are modeled by Petri nets [14]. Centralized controller design to enforce boundedness, liveness, and reversibility was considered in [3]. Here, we use overlapping decompositions and the approach of [3] to design decentralized controllers to enforce these three properties simultaneously. Furthermore, since deadlock does not occur in a live Petri net [15], the controller also avoids deadlock.

2 Preliminaries

2.1 Petri nets and supervisory control

A Petri net is a tuple $G(P, T, N, O, m_0)$, where P is the set of *places*, T is the set of *transitions*, $N : P \times T \to \mathcal{N}$ is the *input matrix* that specifies the *weights* of the *arcs* directed from places to transitions, $O : P \times T \to \mathcal{N}$ is the *output matrix* that specifies the weights of the arcs directed from transitions to places and m_0 is the *initial marking*. Here, $\mathcal{N}$ is the set of natural numbers. $M : P \to \mathcal{N}$ is a *marking vector*, $M(p)$ indicates the number of *tokens* assigned by marking M to place p. A transition $t \in T$ is *enabled* if and only if $M(p) \geq N(p, t)$ for all $p \in P$. An enabled transition t may *fire* at M, yielding the new marking vector:

$$M'(p) = M(p) + O(p, t) - N(p, t) \ , \quad \forall p \in P \qquad (1)$$

A *firing sequence* g is a sequence of enabled transitions. A marking M' is said to be *reachable* from M if there exist a firing sequence starting from M (i.e., the first transition of the sequence fires at M) and yielding M' (i.e., the final transition of the sequence yields M'). The set denoted by $R(G, M)$ is the set of all marking vectors reachable from M. For a Petri net G, we let $\rho(M, g)$ to denote the *transition function*, which gives the yielded marking when the sequence g fires starting from M (ρ is in fact a partial function, since it is not defined if g contains transitions which are not enabled). We also let $\mathcal{E}(M)$ to denote the set of transitions which are enabled at M.

Some important properties for a Petri net are boundedness, reversibility, liveness, and deadlock freeness. For a vector $K : P \to \mathcal{N}$, G is said to be K*-bounded* if $M(p) \leq K(p)$, $\forall p \in P$, $\forall M \in R(G, m_0)$. G is said to be *bounded* if it is K-bounded for some $K : P \to \mathcal{N}$. G is said to be *reversible* if $m_0 \in R(G, M)$, $\forall M \in R(G, m_0)$. G is said to be *live* if, for every $M \in R(G, m_0)$ and for every $t \in T$, there exists a firing sequence g such that t can fire at $\rho(M, g)$. *Deadlock* is said to occur in a Petri net if there exists $M \in R(G, m_0)$ such that no transition $t \in T$ can fire at M.

Supervisory control can be used to guarantee any one of the properties mentioned above. A supervisory controller can disable any transition $t \in T$ depending on the present state M of the Petri net. Such a controller can be described by the *controller function*, $c : R(G, m_0) \times T \to \{0, 1\}$. Here, $c(M, t) = 0$ indicates that $t \in \mathcal{E}(M)$ is disabled by the controller at state M and $c(M, t) = 1$ indicates that $t \in \mathcal{E}(M)$ is enabled by the controller. If, for some $M \in R(G, m_0)$, $t \in T \setminus \mathcal{E}(M)$, then $c(M, t)$ need not be defined, since t is not enabled by the Petri net (thus, c is in fact a partial function). In [3] algorithms were presented to design a controller to enforce boundedness, reversibility, and liveness at the same time in an ordinary Petri net with weighted arcs. These algorithms are presented, using psudo-codes, in the appendix of the present paper for completeness. In these algorithms, for a set S, $|S|$ denotes the number of elements of S, and $[S]_i$ denotes the i^{th} element of S ($i = 1, 2, \ldots, |S|$). The set $\mathcal{E}(M)$, for $M \in R(G, m_0)$, is obtained using Algorithm II. The marking $\rho(M, t)$, for $M \in R(G, m_0)$ and $t \in \mathcal{E}(M)$, is obtained using (1). The main algorithm (Algorithm I) requires the network definition G and the bound vector κ. Main Algorithm calls Algorithms III–VI in order to construct the κ-bounded reachability set, R_B, the κ-bounded reversible set, R_R, to determine whether the κ-bounded reversible net is consistent (a controller which enforces both liveness and reversibility exists if and only if the given Petri net is consistent [3]), and finally to construct the controller function whenever there exists a controller which satisfy all the desired properties (for details, see [3]). It was shown in [3] that the proposed algorithms always find a controller to enforce these properties whenever it is possible. Furthermore, the controller obtained is the least restrictive controller among all controllers which enforce these properties simultaneously.

2.2 Inclusion principle, overlapping decompositions, and expansions

Inclusion principle provides the theoretical framework for controller design using overlapping decompositions. Inclusion principle for Petri nets was introduced in [2]. Consider two Petri nets $G(P, T, N, O, m_0)$ and $\tilde{G}(\tilde{P}, \tilde{T}, \tilde{N}, \tilde{O}, \tilde{m}_0)$. Let $\rho(\cdot, \cdot)$ denote the transition function of G and $\tilde{\rho}(\cdot, \cdot)$ denote the transition function of $\tilde{G}$. Then, inclusion relationship between $\tilde{G}$ and G is defined as follows.

Definition 1 [2]: $\tilde{G}$ *includes* G, if there exist matrices Q and V which satisfy the following conditions.

(i) $QV = I$, where I indicates the identity matrix,

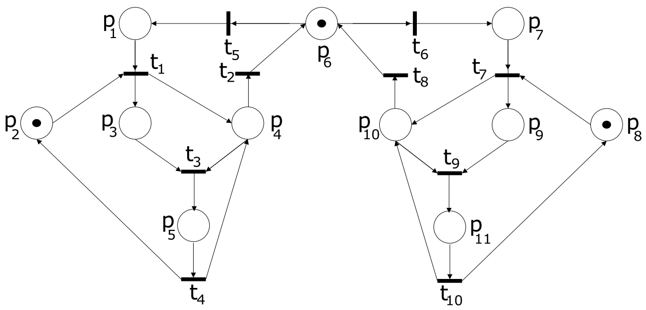

Figure 1: Original Petri net

(ii) $\tilde{m}_0 = V m_0$,

(iii) for each $\tilde{M} \in R(\tilde{G}, \tilde{m}_0)$, $\tilde{M}' \in R(\tilde{G}, \tilde{M})$ only if $Q\tilde{M}' \in R(G, Q\tilde{M})$,

(iv) for each $M \in R(G, m_0)$, there exists $\tilde{M} \in R(\tilde{G}, \tilde{m}_0)$ such that $M = Q\tilde{M}$.

In [2], it was shown that the properties of boundedness, liveness, and reversibility, among others, are carried over from the including net to the included net:

Theorem 1 [2]: Let $\tilde{G}$ include G. Then,

(i) G is bounded if $\tilde{G}$ is bounded.

(ii) G is reversible if $\tilde{G}$ is reversible.

(iii) supposing that for each $t \in T$, there exists a $\tilde{t} \in \tilde{T}$ such that $[O - N]_t = Q[\tilde{O} - \tilde{N}]_{\tilde{t}}$, where $[\cdot]_t$ indicates the column of the matrix $\cdot$ that corresponds to t, G is live if $\tilde{G}$ is live.

Overlapping decompositions and expansions of Petri nets were first considered in [2], where a topological approach was introduced. In this approach, overlapping subnets of a Petri net are first identified by examining the topological structure of the Petri net. These subnets (from here on called *Petri subnets* (PSNs)) are identified such that the only interconnection between the subnets are through the overlapping part, i.e., no arc should be directed from one transition/place in one subnet to a place/transition in another subnet unless at least one of these transitions/places is in the overlapping part of the two subnets. For example, for the Petri net shown in Figure 1, an overlapping decomposition may be obtained as shown in Figure 2.

In the approach of [2], once an overlapping decompositon of the original Petri net is obtained, in order to obtain disjoint subnets, the overlappingly decomposed Petri net is *expanded* by repeating the places, the arcs, and the transitions in the overlapping part (e.g., p_6 in Figure 2 is repeated as p_{6a} and p_{6b} in Figure 3). Transitions (such as t_x and t_y in Figure 3) which interconnect the repeated places through arcs with unity weights are also introduced. As a result, an expanded Petri net (EPN), $\tilde{G}(\tilde{P}, \tilde{T}, \tilde{N}, \tilde{O}, \tilde{m}_0)$, which consists of S disjoint PSNs, is obtained from an original Petri net (OPN), $G(P, T, N, O, m_0)$, which was decomposed into S overlapping PSNs. The initial marking, $\tilde{m}_0$, of the EPN is obtained by assigning all the tokens in a place in the overlapping part of the OPN to one of the

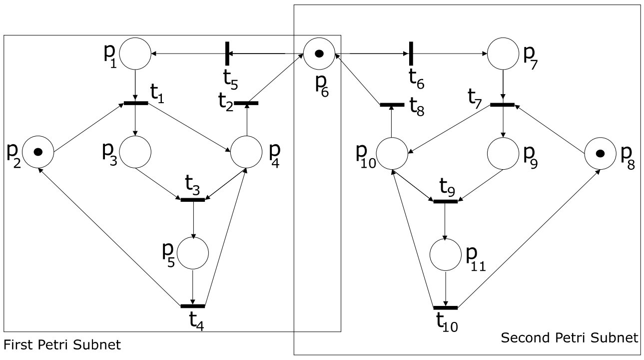

Figure 2: Overlappingly decomposed Petri net

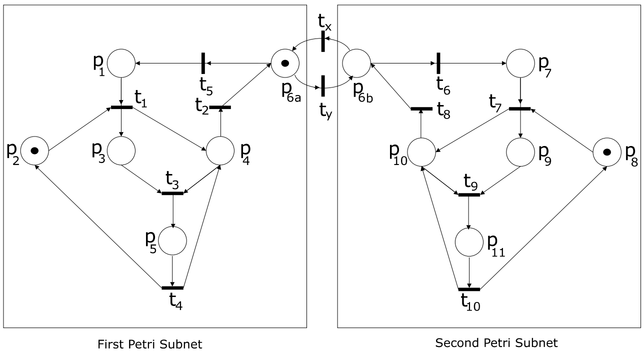

Figure 3: Expanded Petri net

corresponding repeated places (see [2] for details). For example, for the OPN given in Figure 2, the EPN shown in Figure 3 is obtained.

It was shown in [2] that the EPN includes the OPN. Furthermore, it was shown that the additional condition in part (iii) of Theorem 1 (for each $t \in T$, there exists a $\tilde{t} \in \tilde{T}$ such that $[O - N]_t = Q[\tilde{O} - \tilde{N}]_{\tilde{t}}$) is also satisfied. Therefore, by Theorem 1, boundedness, liveness, and reversibility of the EPN respectively implies the boundedness, liveness, and reversibility of the OPN.

3 Main results

In this section we consider decentralized supervisory controller design based on overlapping decompositions of Petri nets to enforce boundedness, liveness, and reversibility. For nontriviality, we assume that the OPN, $G(P, T, N, O, m_0)$, satisfies the following.

Assumption 1: For any $p \in P$, there exists at least one $t \in T$, such that $O(p, t) \geq 1$.

In our approach, we first consider each decoupled PSN separately. To design a controller for the k^{th} PSN, we first determine an initial marking, m_{k_0}, for that PSN such that m_{k_0} is the part of a valid initial marking for the EPN that corresponds to the k^{th} PSN and that there exists a controller which enforces boundedness, liveness, and reversibility for that PSN starting at that initial condition (we assume that such a choice exists for each PSN). We then design a controller for the k^{th} PSN, using the approach of [3], and denote the corresponding controller function by c_k.

Once a controller is obtained for each PSN, we let the controller function for the EPN be described as

$$\tilde{c}(\tilde{M}, \tilde{t}) = \begin{cases} c_j(M_j, \tilde{t}), & \text{if } \tilde{t} \in \mathcal{E}(M_j) \text{ and} \\ & \quad M_j \in R(G_{j_c}, m_{j_0}) \\ 1, & \text{if } \tilde{t} \in \mathcal{T} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

$\forall \tilde{M} \in R(\tilde{G}, \tilde{m}_0)$, $\forall \tilde{t} \in \tilde{T}$, where G_{j_c} denotes the j^{th} PSN under the control c_j, $\mathcal{T}$ is the set of transitions that were introduced between the repeated places (e.g., for the example shown in Figure 3, $\mathcal{T} = \{t_x, t_y\}$) and M_j is the part of the marking vector $\tilde{M}$ which corresponds to the j^{th} PSN.

Theorem 2: The controller (2) enforces boundedness, liveness, and reversibility in the EPN.

Proof: *a) Boundedness:* Controller c_j guarantees that $M(p) \leq \kappa_j(p)$, $\forall p \in P_j$, $\forall M \in R(G_{j_c}, m_{j_0})$, $\forall j \in \{1, \dots S\}$, where P_j is the set of places of the j^{th} PSN, κ_j is the bound vector chosen for the j^{th} PSN, and S is the number of PSNs. For a place $\tilde{p} \in \tilde{P}$, let $\mu(\tilde{p})$ denote the index of the PSN for which $\tilde{p}$ belongs. Also let $\Pi(\tilde{p})$ denote the place p in the OPN which corresponds to $\tilde{p} \in \tilde{P}$ and, for a place $p \in P$, let $\phi_P(p)$ denote the set of places in the EPN which correspond to the place p. Note that controller (2) allows firing $\tilde{t} \in \bar{T} := \tilde{T} \setminus \mathcal{T}$ at $\tilde{M}$, only if $M_j \in R(G_{j_c}, m_{j_0})$ and $\tilde{t} \in \mathcal{E}(M_j)$. This guarantees that the number of tokens in a place $\tilde{p} \in P_j$, such that $\Pi(\tilde{p})$ is not in the overlapping part, can not exceed $\kappa_j(\tilde{p})$. If $\Pi(\tilde{p})$ is in the overlapping part, then the number of tokens in $\tilde{p}$ can not exceed $\max_{\bar{p} \in \phi_P(\Pi(\tilde{p}))} \kappa_{\mu(\bar{p})}(\bar{p})$. Therefore, the EPN is $\tilde{\kappa}$-bounded, where $\tilde{\kappa}(\tilde{p}) = \max_{\bar{p} \in \phi_P(\Pi(\tilde{p}))} \kappa_{\mu(\bar{p})}(\bar{p})$.

b) Reversibility: Consider $\tilde{M} = [M_1^T \mid M_2^T \mid \dots \mid M_S^T]^T \in R(\tilde{G}_c, \tilde{m}_0)$, where superscript T denotes the transpose and $\tilde{G}_c$ denotes the EPN under the controller (2). Then, either there exists a $k \in \{1, \dots S\}$ such that $M_k \in R(G_{k_c}, m_{k_0})$ (in which case we take $\tilde{M}' = \tilde{M}$) or there exists a firing sequence $\tilde{g}_0$, consisting of transitions that belong to $\mathcal{T}$, such that $M'_k \in R(G_{k_c}, m_{k_0})$, for some $k \in \{1, \dots S\}$, where M'_k is the part of $\tilde{M}' = \tilde{\rho}(\tilde{M}, \tilde{g}_0)$ which corresponds to the k^{th} PSN. Then, since G_{k_c} is reversible, there exists a firing sequence $\tilde{g}$, consisting of transitions that belong to the set of transitions of the k^{th} PSN, T_k, such that

$$\tilde{M}'' = \tilde{\rho}(\tilde{M}, \tilde{g}) = [{M'_1}^T \mid {M'_2}^T \mid \dots \mid {M'_{k-1}}^T \mid {m_{k_0}}^T \mid {M'_{k+1}}^T \mid \dots \mid M_S^T]^T$$

Note that $\tilde{M}'' \in R(\tilde{G}_c, \tilde{m}_0)$. Thus, repeating the above procedure S times (for a different $k \in \{1, \dots, S\}$ each time) we arrive to a state from which $\tilde{m}_0$ can be reached by a firing sequence consisting of transitions that belong to $\mathcal{T}$. Thus, $\tilde{G}_c$ is reversible.

c) Liveness: Since $\tilde{G}_c$ is reversible, we only need to show that, for any $\tilde{t} \in \tilde{T}$, there exists a firing sequence $\tilde{g}$ for $\tilde{G}_c$ such that $\tilde{t}$ may fire at $\tilde{\rho}(\tilde{m}_0, \tilde{g})$. We note that either $\tilde{t} \in \bar{T}$ or $\tilde{t} \in \mathcal{T}$.

We first consider the case $\tilde{t} \in \bar{T}$. In this case $\tilde{t} \in T_k$ for some $k \in \{1, \dots, S\}$. Note that, either the part of $\tilde{m}_0$ that corresponds to the k^{th} PSN is equal to m_{k_0} or there exists a firing sequence $\tilde{g}_0$, consisting of transitions that belong to $\mathcal{T}$, such that the part of $\tilde{\rho}(\tilde{m}_0, \tilde{g}_0)$ that corresponds to the k^{th} PSN is equal to m_{k_0}. Now, since G_{k_c} is live, there exists a firing sequence $\tilde{g}_1$, consisting of transitions that belong to T_k, such that $\tilde{t}$ may fire at $\tilde{\rho}(\tilde{\rho}(\tilde{m}_0, \tilde{g}_0), \tilde{g}_1) = \tilde{\rho}(\tilde{m}_0, \tilde{g}_0\tilde{g}_1)$.

Next, we consider the case $\tilde{t} \in \mathcal{T}$. In this case there exists exactly one $\tilde{p} \in \tilde{P}$ such that $\tilde{N}(\tilde{p}, \tilde{t}) = 1$ and $\tilde{N}(\tilde{p}', \tilde{t}) = 0$, for all $\tilde{p}' \neq \tilde{p}$. Note that $\tilde{p} \in P_k$ for some $k \in \{1, \ldots, S\}$. Let us first assume that there exists a $t_1 \in T_k$ for which $\tilde{O}(\tilde{p}, t_1) \geq 1$. Since $t_1 \in \bar{T}$, by the above argument, there exists a firing sequence $\tilde{g}$ for $\tilde{G}_c$, such that t_1 may fire at $\tilde{\rho}(\tilde{m}_0, \tilde{g})$. Firing t_1, however, places at least one token in $\tilde{p}$, which enables $\tilde{t}$. Thus, $\tilde{t}$ may fire at $\tilde{\rho}(\tilde{m}_0, \tilde{g}t_1)$. If there does not exist a $t_1 \in T_k$ for which $\tilde{O}(\tilde{p}, t_1) \geq 1$, then, by Assumption 1, there exists $\tilde{p}_1 \in P_l$, for some $l \in \{1, \ldots, S\} \setminus \{k\}$, such that $\Pi(\tilde{p}_1) = \Pi(\tilde{p})$ and there exists a $t_1 \in T_l$ for which $\tilde{O}(\tilde{p}_1, t_1) \geq 1$. Since $t_1 \in \bar{T}$, by the above argument, there exists a firing sequence $\tilde{g}$ for $\tilde{G}_c$, such that t_1 may fire at $\tilde{\rho}(\tilde{m}_0, \tilde{g})$. Firing t_1, however, places at least one token in $\tilde{p}_1$, which enables the transition $t_2 \in \mathcal{T}$ which connects $\tilde{p}_1$ to $\tilde{p}$. Thus, $\tilde{t}$ may fire at $\tilde{\rho}(\tilde{m}_0, \tilde{g}t_1t_2)$. This proves the liveness of $\tilde{G}_c$. ∎

Once we obtain a controller for the EPN, we can obtain a controller for the OPN by *contracting* the controller for the EPN. To define this contraction, for $M \in R(G, m_0)$, we first define $\Delta_M := \left\{ \tilde{M} \in R(\tilde{G}, \tilde{m}_0) \mid M(p) = \sum_{\tilde{p} \in \phi_P(p)} \tilde{M}(\tilde{p}) \right\}$, where $\phi_P(p)$ denotes the set of places in the EPN which correspond to the place $p \in P$ in the OPN. We also let $\phi_T(t)$ denote the set of transitions in the EPN which correspond to the transition $t \in T$ in the OPN. We now define the contraction of the controller function $\tilde{c}$ of the EPN as

$$c(M, t) = \bigvee_{\tilde{M} \in \Delta_M} \ \bigvee_{\tilde{t} \in \phi_T(t)} \tilde{c}(\tilde{M}, \tilde{t}) \ , \ \forall M \in R(G, m_0), \ \forall t \in T \tag{3}$$

and propose to use this controller for the OPN.

Theorem 3: Let $\tilde{c}$ be a controller for the EPN. Let c be a controller for the OPN such that condition (3) is satisfied. Then the controlled EPN, $\tilde{G}_c$, under the control $\tilde{c}$, includes the controlled OPN, G_c, under the control c.

Proof: First we define a function $\hat{\phi} : P \to \tilde{P}$ as follows:

- If $p \in P$ is not in the overlapping part, then $\hat{\phi}(p) = \phi_P(p)$ (note that, in this case $\phi_P(p)$ is a singleton).
- If $p \in P$ is in the overlapping part and $m_0(p) = 0$, then we arbitrarily choose one $\tilde{p} \in \phi_P(p)$ and let $\hat{\phi}(p) = \tilde{p}$.
- If $p \in P$ is in the overlapping part and $m_0(p) \neq 0$, then we choose $\tilde{p} \in \phi_P(p)$ which satisfies $\tilde{m}_0(\tilde{p}) = m_0(p)$ and let $\hat{\phi}(p) = \tilde{p}$.

Now, we define matrices Q and V as follows

$$Q(p, \tilde{p}) = \begin{cases} 1\ , & \text{if } p = \Pi(\tilde{p}) \\ 0\ , & \text{otherwise} \end{cases}$$

$$V(\tilde{p}, p) = \begin{cases} 1\ , & \text{if } \tilde{p} = \hat{\phi}(p) \\ 0\ , & \text{otherwise} \end{cases}$$

It is easy to check that conditions (i) and (ii) of Definition 1 are satisfied by the above choice of Q and V. To show that condition (iii) is satisfied, let $\tilde{g}$ be a firing sequence for $\tilde{G}_c$ starting from some $\tilde{M} \in R(\tilde{G}_c, \tilde{m}_0)$ and construct a sequence g for G_c as follows

1) Let $\tilde{t}$ denote the first transition in $\tilde{g}$.
2) If $\tilde{t} \in \bar{T}$, include the transition in the OPN which corresponds to $\tilde{t}$ in g. If $\tilde{t} \in \mathcal{T}$, skip to the next step.
3) If $\tilde{t}$ is the last transition in $\tilde{g}$ stop. Otherwise, let $\tilde{t}$ be the next transition in $\tilde{g}$ and go back to step 2.

Then

i) g is a valid firing sequence for G_c (i.e., each transition in g is enabled) starting from $M \in R(G_c, m_0)$, where M is such that $M(p) = \sum_{\tilde{p} \in \phi_P(p)} \tilde{M}(\tilde{p})$, $\forall p \in P$, and

ii) if $\tilde{M}' = \tilde{\rho}(\tilde{M}, \tilde{g})$, then $M' = \rho(M, g)$ is such that $M'(p) = \sum_{\tilde{p} \in \phi_P(p)} \tilde{M}'(\tilde{p})$, $\forall p \in P$.

Condition (iii) of Definition 1 now follows form this result.

To show that condition (iv) of Definition 1 is also satisfied, let g be a firing sequence for G_c starting from m_0 and construct a sequence $\tilde{g}$ for $\tilde{G}_c$ as follows

1) Let t denote the first transition in g. If there exists a $\tilde{t} \in \phi_T(t)$ which is enabled at $\tilde{m}_0$, include $\tilde{t}$ as the first element of $\tilde{g}$ and skip to the next step. Otherwise, by the choice of the initial marking $\tilde{m}_0$, there must exist $\tilde{t}_1, \tilde{t}_2, \ldots, \tilde{t}_l, \in \mathcal{T}$, such that at least one $\tilde{t} \in \phi_T(t)$ is enabled after firing $\tilde{t}_1\tilde{t}_2 \ldots \tilde{t}_l$. Include $\tilde{t}_1\tilde{t}_2 \ldots \tilde{t}_l\tilde{t}$ as the first $l+1$ elements of $\tilde{g}$.
2) Let t' be the transition that follows t in g. If there exists a $\tilde{t}' \in \phi_T(t')$ which is enabled, include $\tilde{t}'$ in $\tilde{g}$ as the next element and skip to the next step. Otherwise, there must exist $\tilde{t}_1, \tilde{t}_2, \ldots, \tilde{t}_l, \in \mathcal{T}$, such that at least one $\tilde{t}' \in \phi_T(t')$ is enabled after firing $\tilde{t}_1\tilde{t}_2 \ldots \tilde{t}_l$. Include $\tilde{t}_1\tilde{t}_2 \ldots \tilde{t}_l\tilde{t}'$ as the next $l+1$ elements of $\tilde{g}$.
3) If t' is the last transition in g stop. Otherwise, let $t \leftarrow t'$ and go back to step 2.

Then

i) $\tilde{g}$ is a valid firing sequence for $\tilde{G}_c$ starting from $\tilde{m}_0$, and

ii) if $M = \rho(m_0, g)$, then $\tilde{M} = \tilde{\rho}(\tilde{m}_0, \tilde{g})$ is such that $\sum_{\tilde{p} \in \phi_P(p)} \tilde{M}(\tilde{p}) = M(p)$, $\forall p \in P$.

Condition (iv) of Definition 1 then follows. This proves the desired result. ∎

As shown in [2], by the expansion procedure employed, for each $t \in T$, the corresponding $\tilde{t} \in \tilde{T}$ satisfies that $[O - N]_t =$

$Q[\tilde{O} - \tilde{N}]_{\tilde{t}}$, where Q is as chosen in the proof of Theorem 3. Therefore, the following result follows from Theorems 1–3.

Corollary 1: The controller, which is obtained by contracting (as described by (3)) the controller (2), enforces boundedness, liveness, and reversibility in the OPN.

4 Example

Consider the Petri net shown in Figure 1, where all the arcs have unity weights. The overlapping decomposition and expansion of this net are respectively shown in Figures 2 and 3. Note that, although the OPN (shown in Figure 1) is bounded, it is neither live, nor reversible. By taking $m_{1_0} = [0\ 1\ 0\ 0\ 0\ 1]^T$ and $m_{2_0} = [1\ 0\ 1\ 0\ 0\ 0]^T$, each PSN, shown in Figure 3, is also bounded, but neither is live or reversible. Each PSN is, in fact **1**-bounded, where $\mathbf{1} := [1\ 1\ 1\ 1\ 1\ 1]^T$. Therefore, we run the algorithms, presented in the Appendix, for each PSN by choosing $\kappa = \mathbf{1}$ and obtain

$$c_1(M^*, \tilde{t}) = \begin{cases} 0, & \tilde{t} = t_2 \\ 1, & t \in \mathcal{E}(M^*) \setminus \{t_2\} \end{cases}$$

for $M^* = [0\ 0\ 1\ 1\ 0\ 0]^T$, $c_1(M, \tilde{t}) = 1\ ,\ \ \forall \tilde{t} \in \mathcal{E}(M)\ ,\ \forall M \in R_{R_1} \setminus \{M^*\}$ where $R_{R_1} = R(G_1, m_{1_0}) \setminus \{[0\ 0\ 1\ 0\ 0\ 1]^T\}$,

$$c_2(M^\dagger, \tilde{t}) = \begin{cases} 0, & \tilde{t} = t_8 \\ 1, & t \in \mathcal{E}(M^\dagger) \setminus \{t_8\} \end{cases}$$

for $M^\dagger = [0\ 0\ 0\ 1\ 1\ 0]^T$, and $c_2(M, \tilde{t}) = 1\ ,\ \ \forall \tilde{t} \in \mathcal{E}(M)\ ,\ \forall M \in R_{R_2} \setminus \{M^\dagger\}$ where $R_{R_2} = R(G_2, m_{2_0}) \setminus \{[1\ 0\ 0\ 1\ 0\ 0]^T\}$.

A controller for the EPN is now obtained using (2). A controller for the OPN is in turn obtained by (3). This final controller enforces $\tilde{\kappa}$-boundedness, liveness, and reversibility in the OPN, where $\tilde{\kappa} = [1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1]^T$.

5 Conclusion

We considered decentralized controller design for Petri nets using overlapping decompositions to enforce boundedness, liveness, and reversibility. Since deadlock does not occur in a live Petri net [15], the controller designed using the proposed approach also avoids deadlock. In the proposed approach, a controller is first designed for each PSN. These controllers are then combined to obtain a controller for the EPN, and the controller for the EPN is contracted to obtain a controller for the OPN. Since, as discussed in [2], each PSN is smaller than the OPN, the number of states (reachable marking vectors) of each PSN is, in general, much smaller than the number of states of the OPN. For example, each decoupled PSN in Figure 3 has 7 states, whereas the OPN, shown in Figure 1, has 24 states. On noting that, the complexity of controller design for a Petri net is proportional to the number of its states, using the proposed approach, controller design may be simplified considerably. For the example given in this paper, for instance, the ratio of the complexity of the proposed design approach, to the complexity of a centralized design is $7/24 = 0.292$ (in this example the two PSNs are identical, thus the controller designed for one can also be used for the other after a re-indexing of places and transitions). This ratio may be much smaller for a more complex Petri net. There is, of course, some overhead in the proposed approach for obtaining a usefull decomposition and for combining and contracting decentralized controllers. The ease of the actual design stage, however, may heavily overweight this overhead. Another advantage of the proposed approach is that, it results in an *overlappingly decentralized* controller (where for controlling a transition which belongs to the k^{th} PSN, one needs to know only the number of tokens in places which belong to the k^{th} PSN). One disadvantage of the proposed approach, on the other hand, is that, it may produce a conservative controller, unnecessarily disabling some transitions. However, such a price must be paid in any decentralized design approach.

Appendix

Algorithm I: Main algorithm:

Main [G, κ]
$< R_B >$ = Bounded-Set[G, κ]
If $R_B == \emptyset$ Then
  "can not design a controller"
  Exit Main
End
$< R_R >$ =Reversible-Set[G, R_B]
$<$ check $>$= Consistent-Test[G, R_R]
  If check == "$not\ consistent$" Then
    "can not design a controller"
    Exit Main
  Else
    For $i = 1$ to $|R_R|$
      $\mathcal{T} = \mathcal{E}([R_R]_i)$
      For $j = 1$ to $|\mathcal{T}|$
        $c([R_R]_i, [\mathcal{T}]_j)$ =Fire-Control[R_R, $[R_R]_i$, $[\mathcal{T}]_j$]
      End
    End
  End

Algorithm II: Algorithm to determine the set of enabled transitions:

subfunction $\mathcal{E}(M)$
$\mathcal{T} = \emptyset$
For $i = 1$ to $|T|$
  For $j = 1$ to $|P|$
    If $M([P]_j) < N([P]_j, [T]_i)$ Then
      Go To Break
    End
  End
  $\mathcal{T} \leftarrow \mathcal{T} \cup \{[T]_i\}$
  Break: Continue
End
Return $\mathcal{T}$

Algorithm III: Algorithm to construct the κ-bounded reachability set:

Bounded-Set [G, κ]
If $m_0 \leq \kappa$ Then
  set $R_B := \{m_0\}$
Else
  set $R_B := \emptyset$
  Return R_B
End
set $R_T := \{m_0\}$
Do Loop Bounded
  $R'_T = \emptyset$
  For $i = 1$ to $|R_T|$
    $\mathcal{T} = \mathcal{E}([R_T]_i)$
    For $j = 1$ to $|\mathcal{T}|$
      $\hat{M} = \rho([R_T]_i, [\mathcal{T}]_j)$
      If $\hat{M} \leq \kappa$
        $R'_T \leftarrow R'_T \cup \{\hat{M}\}$
      End
    End
  End
  If $R'_T \subset R_B$ Then
    Exit Loop Bounded
  End
  $R_T = R'_T$
  $R_B \leftarrow R_B \cup R'_T$
Loop Bounded
Return R_B

Algorithm IV: Algorithm to construct the κ-bounded reversible set:

Reversible-Set[G, R_B]
set $R' := R_B \setminus \{m_0\}$
set $S_R := \{m_0\}$
Do Loop Reversible
  $R = R'$
  $Flag = 0$
  For $i = 1$ to $|R|$
    $\mathcal{T} = \mathcal{E}([R]_i)$
      For $k = 1$ to $|\mathcal{T}|$
        $\hat{M} = \rho([R]_i, [\mathcal{T}]_k)$
        If $\hat{M} \in S_R$ Then
          $S_R \leftarrow S_R \cup \{[R]_i\}$
          $R' \leftarrow R' \setminus \{[R]_i\}$
          $Flag = 1$
          Go To Break
        End
      End
  Break: Continue
  End
  If $Flag == 0$ Then
    Exit Loop Reversible
  End
Loop Reversible
Return S_R

Algorithm V: Algorithm to determine consistency:

Consistent-Test [G, R_R]
data="*not consistent*"
$\mathcal{T} = T$
For $i = 1$ to $|R_R|$
  $J = \mathcal{E}([R_R]_i)$
  For $j = 1$ to $|J|$
    If $\rho([R_R]_i, [J]_j) \in R_R$ Then
      $\mathcal{T} \leftarrow \mathcal{T} \setminus \{[J]_j\}$
      If $\mathcal{T} == \emptyset$ Then
        data="*consistent*"
        Go To Break
      End
    End
  End
End
Break: Return data

Algorithm VI: Algorithm to determine the controller function:

Fire-Control [R_R, M, t]
If $\rho(M, t) \in R_R$ Then
  r=1
Else
  r=0
End
Return r

Acknowledgement

This work was supported by Anadolu University through Research Project 020222.

References

[1] A. Aybar and A. İftar. Decentralized control design for interconnected discrete–event systems. In *Preprints of the 9th IFAC Symposium on Large Scale Systems*, pp. 415–418, Bucharest, Romania, July 2001.

[2] A. Aybar and A. İftar. Overlapping decompositions and expansions of Petri nets. *IEEE Transactions on Automatic Control*, **47**, pp. 511–515, (2002).

[3] A. Aybar and A. İftar. Controller design to enforce boundedness, liveness, and reversibility in Petri nets. In *Preprints of the 7th IFAC Workshop on Intelligent Manufacturing Systems*, pp. 199–204, Budapest, Hungary, April 2003.

[4] A. Aybar and A. İftar. Decentralized supervisory controller design to avoid deadlock in Petri nets. *International Journal of Control*, (2003). (To appear).

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

[6] C. Haoxun. Net structure and control logic synthesis of controlled Petri nets. *IEEE Transactions on Automatic Control*, **43**, pp. 1446–1450, (1998).

[7] L. E. Holloway and B. H. Krogh. Synthesis of feedback control logic for a class of controlled Petri nets. *IEEE Transactions on Automatic Control*, **35**, pp. 514–523, (1990).

[8] A. İftar and Ü. Özgüner. Decentralized LQG/LTR controller design for interconnected systems. In *Proceedings of the American Control Conference*, pp. 1682–1687, Minneapolis, MN, June 1987.

[9] A. İftar and Ü. Özgüner. Overlapping decompositions, expansions, contractions, and stability of hybrid systems. *IEEE Transactions on Automatic Control*, **43**, pp. 1040–1055, (1998).

[10] M. Ikeda and D. D. Šiljak. Overlapping decompositions, expansions, and contractions of dynamic systems. *Large Scale Systems*, **1**, pp. 29–38, (1980).

[11] M. Ikeda and D. D. Šiljak. Overlapping decentralized control with input, state, and output inclusion. *Control Theory and Advanced Technology*, **2**, pp. 155–172, (1986).

[12] R. Kumar and L. E. Holloway. Supervisory control of Petri net languages. In *Proceedings of the IEEE Conference on Decision and Control*, pp. 1190–1195, Tuscon, AZ, December 1992.

[13] F. Lin and W. M. Wonham. Decentralized control and coordination of discrete-event systems with partial observation. *IEEE Transactions on Automatic Control*, **35**, pp. 1330–1337, (1990).

[14] J. Proth and X. Xie. *Petri Nets: A Tool for Design and Management of Manufacturing Systems*. John Wiley & Sons, West Sussex, 1996.

[15] R. S. Sreenivas. On the existence of supervisory policies that enforce liveness in discrete-event dynamic systems modelled by controlled Petri nets. *IEEE Transactions on Automatic Control*, **42**, pp. 928–945, (1997).