Kybernetika

Aydın Aybar; Altuğ İftar

Decentralized structural controller design for large-scale discrete-event systems modelled by Petri nets

Kybernetika, Vol. 45 (2009), No. 1, 3--14

Persistent URL: http://dml.cz/dmlcz/139999

KYBERNETIKA — VOLUME *45* (2009), NUMBER 1, PAGES 3 – 14

DECENTRALIZED STRUCTURAL CONTROLLER DESIGN FOR LARGE–SCALE DISCRETE–EVENT SYSTEMS MODELLED BY PETRI NETS

Aydin Aybar and Altuğ İftar

A decentralized structural controller design approach for discrete-event systems modelled by Petri nets is presented. The approach makes use of overlapping decompositions. The given Petri net model is first overlappingly decomposed into a number of Petri subnets and is expanded to obtain disjoint Petri subnets. A structural controller is then designed for each Petri subnet to avoid deadlock. The obtained controllers are finally applied to the original Petri net. The proposed approach significantly reduces the computational burden to design the controller. Furthermore, the controller obtained is decentralized and, hence, is easier to implement.

Keywords: large-scale systems, decentralized control, discrete-event systems, Petri nets, overlapping decompositions

AMS Subject Classification: 93A15, 93A14, 93C65

1. INTRODUCTION

Behaviour of many large-scale systems, such as manufacturing systems, communication systems, and transportation systems, can best be described by occurrence of certain events. Such systems are commonly named as *discrete-event systems* (DESs) [9, 14]. A good way to model a DES is by using *Petri nets* [29, 34].

A controller may be required for a DES to avoid undesirable behaviour, such as deadlock (e. g., [5, 12, 13, 31, 33]), or to enforce desirable behaviour, such as liveness, boundedness, or reversibility (e. g., [6, 15, 25]). There are two main control approaches to controller design for DESs modelled by Petri nets: *the forbidden states approach* (e. g., [15, 25, 32, 33]) and *the structural approach* (e. g., [8, 25, 26, 27]). In the forbidden states approach, the reachability set of the given Petri net must be constructed. In the structural approach, it is required to identify certain parts of the Petri net, such as *siphons*, which has certain structural properties. Constructing the reachability set of a large-scale Petri net may be very time consuming [11]. Identifying parts of a Petri net with certain structural properties, on the other hand, also becomes more time consuming as the Petri net gets more complex [10].

For a large-scale complex DES, its Petri net description would also be complex. Therefore, it is more difficult to design a controller for a large-scale complex DES, whether the forbidden states approach or the structural approach is used. As in the case of continuous-state systems, decomposition-based methods for decentralized controller design may be useful to overcome this difficulty. Furthermore, for most large-scale systems there may exist communication constraints, which must be taken into account for controller design. In such a case, decentralized controller design is required. Even if there are no communication constraints, a decentralized design approach may be preferred for ease of design and/or implementation.

A useful decomposition approach for decentralized controller design is *overlapping decompositions*, which was originally introduced for continuous-state systems [7, 16, 17, 18, 19, 21, 22, 23, 28]. This approach was first considered for DESs (as a special case for hybrid systems) in [20]. It was then considered for data-communication networks in [1], for DESs modelled by finite automata in [2] and by Petri nets in [3]. Overlapping decompositions has been used for supervisory controller design for Petri nets to avoid deadlock in [5] and to enforce liveness, boundedness, and reversibility in [6]. In both [5] and [6], however, forbidden states approach was used.

In the present work, using overlapping decompositions and the structural approach, we present a decentralized controller design approach for DESs modelled by Petri nets. Using the presented approach, a controller which avoids deadlock is obtained, whenever it exists. Using the proposed approach, the computational burden to design the controller is significantly reduced compared to the centralized approach. Furthermore, the controller obtained is decentralized and, hence, is easier to implement in most cases.

2. PRELIMINARIES

2.1. Petri nets and structural control

A *Petri net* is a five tuple $G = (P, T, N, O, m_0)$, where P is the set of *places*, T is the set of *transitions*, $N : P \times T \to \{0, 1\}$ is the *input matrix*, which specifies the arcs directed from places to transitions, $O : P \times T \to \{0, 1\}$ is the *output matrix*, which specifies the arcs directed from transitions to places, and $m_0 : P \to \mathbb{N}$ is the *initial marking*, where $\mathbb{N}$ denotes the set of natural numbers.

$M : P \to \mathbb{N}$ 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 can *fire* at M, yielding the new marking vector:

$$M'(p) = M(p) + O(p, t) - N(p, t) \ , \ \forall\, p \in P \ . \tag{1}$$

The *reachability set* of a Petri net G, denoted by $R(G, m_0)$, is the set of all markings of G, which can be reached by firing a sequence of enabled transitions starting from m_0. An important property for a Petri net is *deadlock freeness*. Deadlock is said to occur in a Petri net G, if there exists $M \in R(G, m_0)$, such that no $t \in T$ is enabled.

The $\bullet$ notation is used to denote the set of places/transitions which preceed/follow a certain transition/place. Specifically, for $p \in P$, $p\bullet := \{t \in T \mid N(p, t) = 1\}$ and $\bullet p := \{t \in T \mid O(p, t) = 1\}$. Similarly, for $t \in T$, $t\bullet := \{p \in P \mid O(p, t) = 1\}$ and $\bullet t := \{p \in P \mid N(p, t) = 1\}$. Also for $S \subset P$, $S\bullet := \bigcup_{p \in S} p\bullet$ and $\bullet S := \bigcup_{p \in S} \bullet p$.

A non-empty set $S \subset P$, is said to be a *siphon* if $\bullet S \subset S\bullet$. A siphon S is said to be a *minimal siphon* if it does not contain any other siphon. A siphon is said to be a *controlled siphon*, if $M(p) \geq 1$ for at least one $p \in S$, $\forall\, M \in R(G, m_0)$. It is known that, for any siphon S, if $m_0(p) \geq 1$ for at least one $p \in S$, then S can be made a controlled siphon by adding a *control place* p_c. The output and input transitions of this control place are respectively defined as follows:

$$p_c\bullet = \left\{ t \in S\bullet \;\middle|\; \sum_{p \in \bullet t \cap S} N(p,t) > \sum_{p \in t\bullet \cap S} O(p,t) \right\} \tag{2}$$

and

$$\bullet p_c = \left\{ t \in \bullet S \;\middle|\; \sum_{p \in \bullet t \cap S} N(p,t) < \sum_{p \in t\bullet \cap S} O(p,t) \right\} . \tag{3}$$

The initial marking of this control place, on the other hand, is taken as

$$m_0^c(p_c) = \sum_{p \in S} m_0(p) - 1 \;, \tag{4}$$

where m_0^c denotes the initial marking of the *controlled Petri net*, G_c, obtained by adding the control places to the given Petri net G.

Furthermore, for a siphon S, if $Q \subset S$ is a controlled siphon, then S is also a controlled siphon. Therefore, it is sufficient to control only the minimal siphons of G. If G contains only controlled siphons, then deadlock does not occur in G. Therefore, a widely used control strategy to avoid deadlock is to add a control place for each uncontrolled minimal siphon, so that the controlled Petri net does not contain any uncontrolled siphons, which guarantees deadlock freenees.

For example, let us consider the Petri net shown in Figure 1, which is borrowed from [4]. In this Petri net, if t_5, t_1, t_2, t_5 or t_6, t_7, t_8, t_6 fire at the initial marking, then deadlock occurs. To design a controller to avoid deadlock, we identify the minimal siphons as $S_1 = \{p_1, p_4, p_5, p_6, p_7, p_{10}, p_{11}\}$, $S_2 = \{p_2, p_3, p_5\}$, $S_3 = \{p_2, p_4, p_5\}$, $S_4 = \{p_8, p_9, p_{11}\}$, and $S_5 = \{p_8, p_{10}, p_{11}\}$. Among those, S_1, S_2, and S_4 are controlled siphons and S_3 and S_5 are uncontrolled siphons. To make S_3 a controlled siphon, we add the control place p_c^1 and to make S_5 a controlled siphon, we add the control place p_c^2. By (2) and (3), $p_c^1\bullet = \{t_2\}$, $\bullet p_c^1 = \{t_4\}$, $p_c^2\bullet = \{t_8\}$, $\bullet p_c^2 = \{t_{10}\}$. By (4), on the other hand, $m_0^c(p_c^1) = m_0^c(p_c^2) = 0$. The resulting controlled Petri net is shown in Figure 2. It can be verified that deadlock does not occur in this Petri net.

2.2. Overlapping decompositions and expansions

Overlapping decompositions and expansions of Petri nets were first introduced in [3]. Overlapping decomposition of a Petri net is obtained by identifying the overlapping subnets (from here on called *Petri subnets* (PSNs)) by an inspection of the Petri net's topological structure. In this work, we allow the *overlapping part* of any two or more PSNs to contain only places (i. e., no transitions are allowed in the overlapping part). Furthermore, the only interconnection between the PSNs must be through the overlapping part, i. e., no arc should be directed from one transition/place in

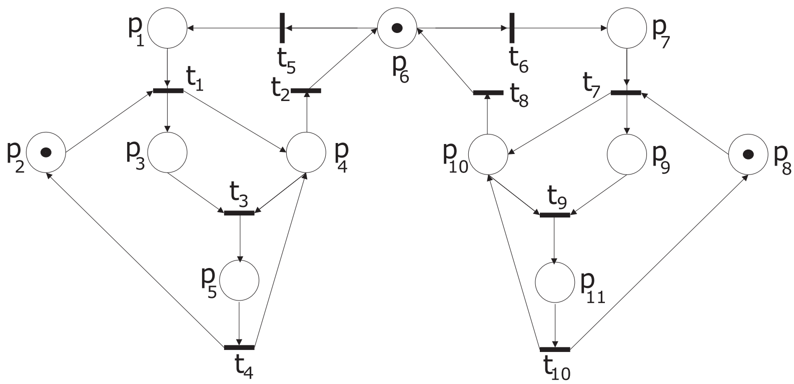

Fig. 1. The example Petri net [4].

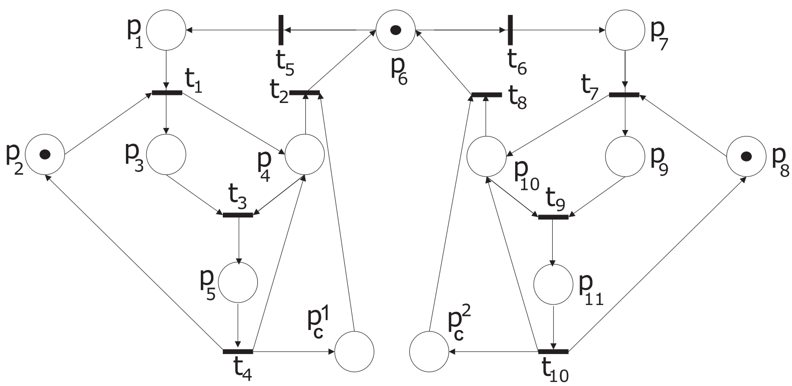

Fig. 2. The controlled Petri net.

one PSN to a place/transition in another PSN unless one of these places is in the overlapping part of the two PSNs. For example, for the example Petri net shown in Figure 1, an overlapping decomposition may be obtained as shown in Figure 3. The overlapping part of the two PSNs contain place p_6 in this example.

Once an overlapping decomposition of the original Petri net is obtained, in order to obtain disjoint subnets, the overlappingly decomposed Petri net is *expanded* as follows:

i) A place in the overlapping part of n PSNs is repeated n times and each repeated place is assigned to a different PSN.

ii) Two transitions with proper arcs are added between any two repeated places, such that each transition, when fire, transfers a token from one repeated place to the other.

iii) Each token which is initially assigned to a place in an overlapping part of the original Petri net is assigned to one of the repeated places corresponding to that place. Number of tokens in any place which is not in any overlapping part remain unchanged.

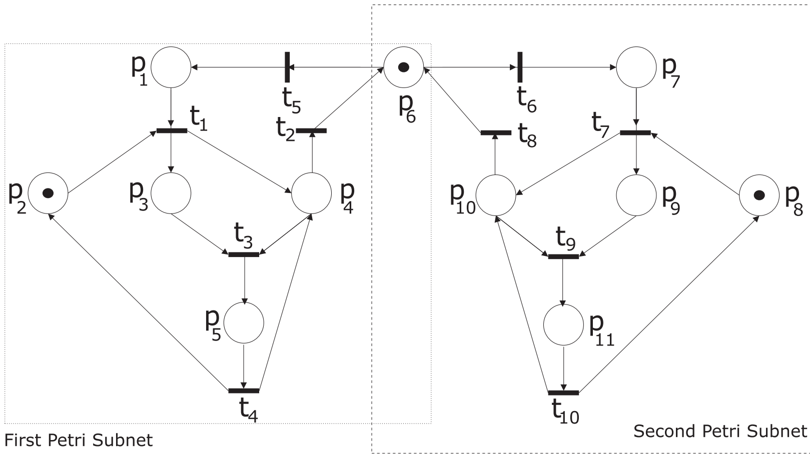

Fig. 3. Overlappingly decomposed example Petri net [4].

As a result of this procedure, an *expanded Petri* net (EPN), $\tilde{G} = (\tilde{P}, \tilde{T}, \tilde{N}, \tilde{O}, \tilde{m}_0)$, which consists of λ disjoint PSNs, is obtained from an original Petri net (OPN), $G = (P, T, N, O, m_0)$, which was decomposed into λ overlapping PSNs. Each place/transition/arc of a PSN corresponds to a place/transition/arc of the OPN. The PSNs are interconnected through the transitions introduced in step (ii) of the above procedure. Step (iii) of the above procedure determines the initial marking, $\tilde{m}_0$, of the EPN. The set of places of the EPN is given by $\tilde{P} := \bigcup_{i=1}^{\lambda} P_i$, where P_i is the set of places of the ith PSN. The set of transitions of the EPN, on the other hand, is given by $\tilde{T} := \left(\bigcup_{i=1}^{\lambda} T_i\right) \bigcup \mathcal{T}$, where T_i is the set of transitions of the ith PSN and $\mathcal{T}$ is the set of transitions between the PSNs (as introduced in step (ii) of the above procedure). For example, for the OPN shown in Figure 1, which is overlappingly decomposed into two PSNs as shown in Figure 3, the above procedure produces the EPN shown in Figure 4. Here, place p_6 is repeated as p_{6a} and p_{6b} and the former is assigned to the first PSN, while the latter is assigned to the second PSN. Two transitions, t_x and t_y, are introduced between these two places. Finally, the token in place p_6 is assigned to p_{6a} in order to define the initial marking $\tilde{m}_0$. The sets of places and transitions of this EPN are respectively given by $\tilde{P} = P_1 \cup P_2$ and $\tilde{T} = T_1 \cup T_2 \cup \mathcal{T}$, where $P_1 = \{p_1, p_2, p_3, p_4, p_5, p_{6a}\}$, $P_2 = \{p_{6b}, p_7, p_8, p_9, p_{10}, p_{11}\}$, $T_1 = \{t_1, t_2, t_3, t_4, t_5\}$, $T_2 = \{t_6, t_7, t_8, t_9, t_{10}\}$, and $\mathcal{T} = \{t_x, t_y\}$.

3. DECENTRALIZED CONTROLLER DESIGN

Consider a given OPN $G = (P, T, N, O, m_0)$, overlappingly decomposed into λ PSNs. Let $\tilde{G} = (\tilde{P}, \tilde{T}, \tilde{N}, \tilde{O}, \tilde{m}_0)$ be the corresponding EPN. Also let $G_k = (P_k, T_k, N_k, O_k, m_{k_0})$ denote the kth PSN, $k = 1, \ldots, \lambda$. The initial marking of G_k is obtained by taking the appropriate part of $\tilde{m}_0$; i. e., $m_{k_0}(p) = \tilde{m}_0(p)$, $\forall\, p \in P_k$.

We define $\Delta : \tilde{P} \to P$ such that, for $\tilde{p} \in \tilde{P}$, $\Delta(\tilde{p})$ denotes the place in P which corresponds to $\tilde{p} \in \tilde{P}$. For example, for the OPN shown in Figure 1 and the EPN

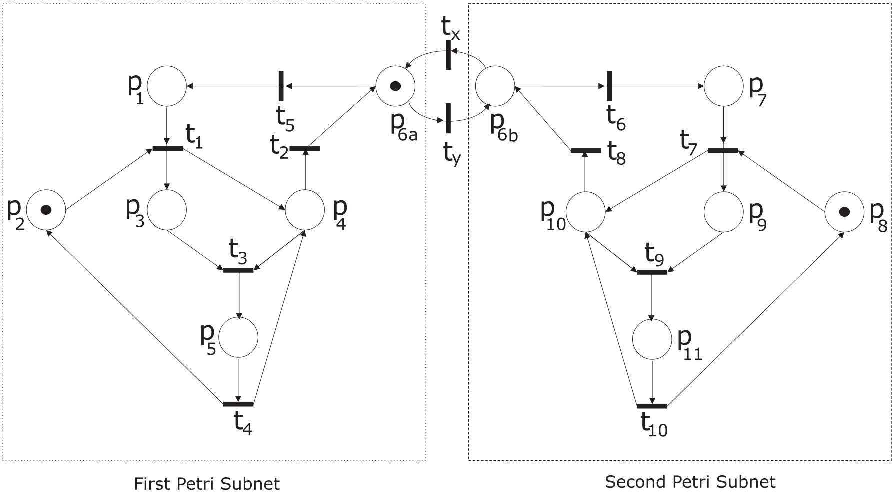

Fig. 4. Expanded Petri net [4].

shown in Figure 4, $\Delta(p_1) = p_1$ and $\Delta(p_{6a}) = \Delta(p_{6b}) = p_6$. We also define $\Theta : P \to 2^{\tilde{P}}$ such that, for $p \in P$, $\Theta(p)$ denotes the set of places in $\tilde{P}$ which corresponds to $p \in P$. For example, for the OPN shown in Figure 1 and the EPN shown in Figure 4, $\Theta(p_1) = \{p_1\}$ and $\Theta(p_6) = \{p_{6a}, p_{6b}\}$. Furthermore, for $S \subset P$ and $\tilde{S} \subset \tilde{P}$, $\Theta(S) := \bigcup_{p \in S} \Theta(p)$ and $\Delta(\tilde{S}) := \bigcup_{p \in \tilde{S}} \{\Delta(p)\}$.

To obtain a decentralized design approach, we first prove the following.

Lemma 1. S is a siphon of the OPN, G, if and only if $\tilde{S} = \Theta(S)$ is a siphon of the EPN, $\tilde{G}$. Furthermore, S is a controlled siphon of G if and only if $\tilde{S}$ is a controlled siphon of $\tilde{G}$.

Proof. Let $\mathcal{T}_S\bullet = \tilde{S}\bullet \cap \mathcal{T}$ and $\bullet\mathcal{T}_S = \bullet\tilde{S} \cap \mathcal{T}$. Then, $\tilde{S}\bullet = S\bullet \cup \mathcal{T}_S\bullet$ and $\bullet\tilde{S} = \bullet S \cup \bullet\mathcal{T}_S$.

Since $\tilde{S} = \Theta(S)$, for any $p \in \tilde{S}$ either $\Delta(p) = p$ (i.e., p is not in any overlapping part, in which case $p\bullet$ and $\bullet p$ does not contain any elements of $\mathcal{T}$) or $\Theta(\Delta(p)) \subset \tilde{S}$ (which implies that for any $t \in \mathcal{T}$, if $t \in p_a\bullet$ for some $p_a \in \tilde{S}$, then there exists $p_b \in \tilde{S}$ such that $t \in \bullet p_b$ and vice versa). This implies that $\mathcal{T}_S\bullet = \bullet\mathcal{T}_S$.

Therefore, $\bullet S \subset S\bullet$ if and only if $\bullet\tilde{S} \subset \tilde{S}\bullet$, which proves that S is a siphon of G if and only if $\tilde{S}$ is a siphon of $\tilde{G}$.

It is known that [3] for any $\tilde{M} \in R(\tilde{G}, \tilde{m}_0)$ there exists $M \in R(G, m_0)$ such that

$$M(p) = \sum_{\tilde{p} \in \Theta(p)} \tilde{M}(\tilde{p}) \qquad \forall\, p \in P\,. \tag{5}$$

Therefore, if $M(p) \geq 1$ for at least one $p \in S$, $\forall\, M \in R(G, m_0)$, then, for all $\tilde{M} \in R(\tilde{G}, \tilde{m}_0)$, $\tilde{M}(\tilde{p}) \geq 1$ for at least one $\tilde{p} \in \tilde{S}$. On the other hand, it is also known that [3] for any $M \in R(G, m_0)$ there exists $\tilde{M} \in R(\tilde{G}, \tilde{m}_0)$ such that (5) holds. Therefore, if $M(p) = 0$, $\forall\, p \in S$, for some $M \in R(G, m_0)$, then there exists

$\tilde{M} \in R(\tilde{G}, \tilde{m}_0)$ such that $\tilde{M}(\tilde{p}) = 0$, $\forall \tilde{p} \in \tilde{S}$. These imply that S is a controlled siphon if and only if $\tilde{S}$ is a controlled siphon. □

Lemma 2. If $\tilde{S}$ is a siphon of the EPN, $\tilde{G}$, then $S_k = \tilde{S} \cap P_k$ is a siphon of the kth PSN, G_k, for all $k \in \kappa(\tilde{S})$, where $\kappa(\tilde{S}) = \{k \mid \tilde{S} \cap P_k \neq \emptyset\}$. Furthermore, if S_k is a controlled siphon of G_k, for all $k \in \kappa(\tilde{S})$, then $\tilde{S}$ is a controlled siphon of $\tilde{G}$.

Proof. Let $\tilde{S}$ be a siphon of $\tilde{G}$ and $S_k = \tilde{S} \cap P_k$. Consider any $t \in \bullet S_k$ for any $k \in \kappa(\tilde{S})$. Since $t \in \bullet S_k$, $t \in T_k$. Furthermore, since $S_k \subset \tilde{S}$, $t \in \bullet\tilde{S}$. Since $\tilde{S}$ is a siphon, this implies $t \in \tilde{S}\bullet$, i.e., $\exists p \in \tilde{S}$ such that $t \in p\bullet$ or $p \in \bullet t$. Since for any $t \in T_k$, $\bullet t \subset P_k$, this implies $t \in S_k\bullet$. Hence, S_k is a siphon of G_k.

If S_k is a controlled siphon of G_k, then it can become empty in $\tilde{G}$ only by a firing of some $t \in \mathcal{T}$. Such a t, however, when fire transfers one token from $p_a \in S_k$ to some $p_b \in \Theta(\Delta(p_a))$. However, such $p_b \in S_l$ for some $l \in \kappa(\tilde{S})$. Therefore, although S_k may become empty for some $k \in \kappa(\tilde{S})$, $\tilde{S}$ never becomes empty. Which implies that $\tilde{S}$ is a controlled siphon of $\tilde{G}$. □

We now propose the following decentralized controller design approach:

Controller Design Algorithm.

Step 1. Obtain an overlapping decomposition and expansion of the given OPN as described in Subsection 2.2.

Step 2. For each disjoint PSN, identify all minimal siphons. If there are any minimal siphons which are not controlled, introduce control places, as explained in Subsection 2.1, to make those siphons controlled. If an uncontrolled siphon is initially empty, design a controller by assuming that there initially exists one token in any one of the places of this siphon which corresponds to a place in the overlapping part of the OPN. If such a siphon does not contain any places which corresponds to a place in the overlapping part of the OPN, then this siphon corresponds to an uncontrolled siphon of the OPN which is initially empty. In this case there exist no controller (even through a centralized design) which can make this siphon a controlled siphon.

Step 3. Apply the control places determined in the above step to the OPN. Note that, since for a control place p_c, determined for the kth PSN, G_k, $p_c\bullet \subset T_k \subset T$ and $\bullet p_c \subset T_k \subset T$, application of these control places to the OPN is straight forward.

The following theorem shows that using the above algorithm, a controlled OPN which has no uncontrolled siphons is obtained, provided only that the uncontrolled minimal siphons of the given OPN are initially non-empty. Note that, if the given OPN has an uncontrolled minimal siphon which is initially empty, then there exists no controller (even through a centralized design) which can produce a controlled OPN without any uncontrolled siphons.

Theorem 1. Assuming that all the uncontrolled minimal siphons of the given OPN are initially non-empty, the Controller Design Algorithm given above produces a controlled OPN which does not have any uncontrolled siphons.

P r o o f . By Lemma 2, if all the PSNs contain only controlled siphons, then the EPN will contain only controlled siphons. Then, by Lemma 1, OPN will also contain only controlled siphons. Since, assuming that all the uncontrolled minimal siphons of the given OPN are initially non-empty, step 2 of the Controller Design Algorithm makes all the siphons of all the PSNs controlled siphons, the desired result follows. □

Since having no uncontrolled siphons for a Petri net ensures deadlock free operation, Theorem 1 also implies the following result.

Corollary 1. Assuming that all the uncontrolled minimal siphons of the given OPN are initially non-empty, the Controller Design Algorithm given above produces a controlled OPN which is deadlock free.

Remark 1. It has been accepted that the number of minimal siphons in a Petri net is exponential with its size [10]. Furthermore, the computational burden to identify each minimal siphon also increases at least with the square of the Petri net size [10]. Therefore, since each PSN is much smaller than the OPN, it is much easier to identify the minimal siphons of each PSN compared to identifying the minimal siphons of the OPN. Hence, it is easier to design a controller using the proposed approach, compared to the centralized approach. Furthermore, the proposed approach produces a decentralized controller, since each control place is connected to the transitions of a given PSN only. Therefore, it may be easier to implement this controller, especially when each PSN corresponds to a different part of a large-scale system, where communication between such parts may be costly. In case when such communication is impossible, the proposed approach directly produces a controller which satisfies this communication constraint. On the other hand, an uncontrolled minimal siphon of a particular PSN may not actually be a part of an uncontrolled minimal siphon of the EPN. In this case, the controller designed for the OPN may unnecessarily restrict certain firings. Therefore, there is a trade-off between ease of design and implementation on the one hand and the performance of the controlled system on the other hand. Such a trade-off, however, always exits in any decentralized controller design approach.

4. EXAMPLE

Consider the OPN shown in Figure 1, which is overlappingly decomposed into two PSNs, as shown in Figure 3. The corresponding EPN is shown in Figure 4. Disjoint PSNs are shown in Figure 5.

Consider the first PSN shown in Figure 5(a). This PSN has three minimal siphons: $S_1^1 = \{p_1, p_4, p_5, p_{6a}\}$, $S_1^2 = \{p_2, p_3, p_5\}$, and $S_1^3 = \{p_2, p_4, p_5\}$. Among those, only S_1^3 is an uncontrolled siphon. To make this siphon a controlled siphon, we add the control place p_c^1. The output and input transitions of this control place are found by using (2) and (3) as: $p_c^1\bullet = \{t_2\}$, $\bullet p_c^1 = \{t_4\}$. The initial marking of this control place, on the other hand, is determined by (4) as $m_{1_0}^c(p_c^1) = 0$. The controlled PSN is shown in Figure 6(a).

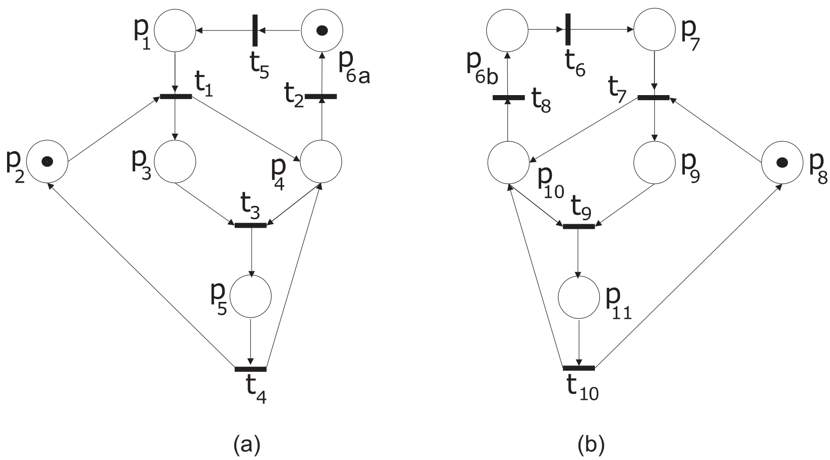

Fig. 5. (a) First PSN; (b) Second PSN.

Next, consider the second PSN shown in Figure 5(b). This PSN has also three minimal siphons: $S_2^1 = \{p_{6b}, p_7, p_{10}, p_{11}\}$, $S_2^2 = \{p_8, p_9, p_{11}\}$, and $S_2^3 = \{p_8, p_{10}, p_{11}\}$. S_2^1 is initially empty. Therefore, according to step 2 of the Controller Design Algorithm, we assume a token in p_{6b}, i. e., we change the initial marking vector as $m_{2_0}(p_{6b}) = 1$. Then, however, S_2^1 becomes a controlled siphon. Hence, there is no need to add a control place for S_2^1. S_2^2 is already a controlled siphon. S_2^3, however, is an uncontrolled siphon. To make S_2^3 a controlled siphon, we add the control place p_c^2. The output and input transitions of this control place are found by using (2) and (3) as: $p_c^2\bullet = \{t_8\}$, $\bullet p_c^2 = \{t_{10}\}$. The initial marking, on the other hand, is determined by (4) as $m_{2_0}^c(p_c^2) = 0$. The controlled PSN is shown in Figure 6(b).

Once we obtain the necessary control places for each disjoint PSN as above, using step 3 of the Controller Design Algorithm, we apply those to the OPN. The resulting controlled OPN is shown in Figure 2. In this example, our decentralized controller design approach produces the same controller as the centralized design approach. However, it was easier to obtain the controller using the decentralized approach, since we considered only two PSNs, each containing only 6 places and 5 transitions, compared to the overall OPN, which contains 11 places and 10 transitions.

Using the MATLAB program developed in [24] on a personal computer with 1 GB RAM and a Core2Due microprocessor running at 1800 MHz, it took 516 milliseconds to find all the minimal siphons for the OPN. On the other hand, it took 46 and 32 milliseconds to find all the minimal siphons for the first and the second PSN, respectively. Therefore, the ratio of the computational time (to find the minimal siphons) of the proposed approach to that of the centralized approach is $(46 + 32)/516 = 0.1512$. That is, the proposed approach is about seven times faster than the centralized approach, as far as identifying the minimal siphons (which is the main computational burden) is concerned.

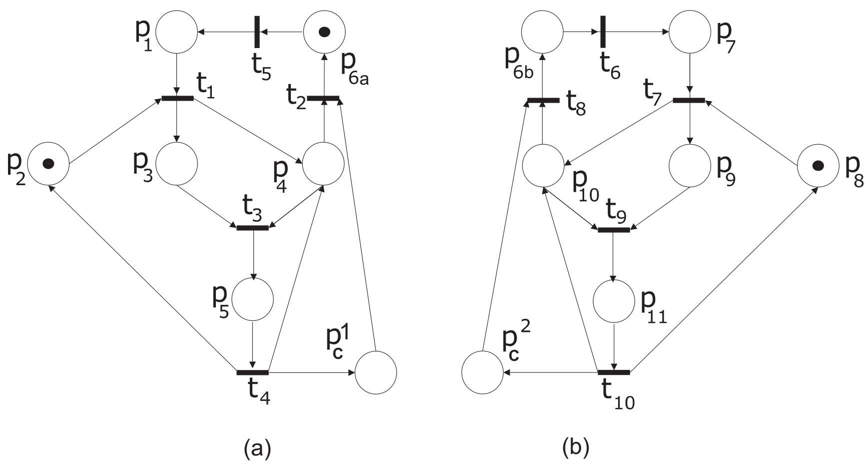

Fig. 6. (a) First controlled PSN; (b) Second controlled PSN.

5. CONCLUSION

We have presented a decentralized structural controller design approach for discrete-event systems modelled by Petri nets. The proposed approach uses overlapping decompositions and produces a decentralized controller which avoids deadlock, whenever it exists. In this approach, the given Petri net model is first overlappingly decomposed into a number of Petri subnets and is expanded to obtain disjoint Petri subnets. A structural controller is then designed for each Petri subnet to avoid deadlock. The obtained controllers are finally applied to the original Petri net. As explained in Remark 1, the proposed approach significantly reduces the computational burden to design the controller. For example, for the example system considered in Section 4, we found that the proposed approach is about seven times faster than the centralized approach. In general, more savings in the computational time is possible for more complex systems. Furthermore, the controller obtained by the proposed approach is decentralized. This controller, as opposed to a centralized controller, requires no communication between subsystems. Therefore, it is easier to implement this controller in general.

Although we have considered controller design for deadlock prevention, the approach of overlapping decompositions may also be used in structural controller design to enforce liveness, boundedness, and/or reversibility. Furthermore, similar approaches may also be developed for special types of Petri nets.

(Received March 28, 2008.)

REFERENCES

[1] B. Ataşlar and A. İftar: Decentralized routing controller design using overlapping decompositions. Internat. J. Control *72* (1999), 1175–1192.

[2] A. Aybar and A. İftar: Decentralized supervisory controller design for discrete-event systems using overlapping decompositions and expansions. Dynamics of Continuous, Discrete and Impulse Systems (Series B) *11* (2004), 553–568.

[3] A. Aybar and A. İftar: Overlapping decompositions and expansions of Petri nets. IEEE Trans. Automat. Control *47* (2002), 511–515.

[4] A. Aybar and A. İftar: Decentralized controller design to enforce boundedness, liveness and reversibility in Petri nets. In: Proc. CD-ROM of the European Control Conference, Cambridge 2003.

[5] A. Aybar and A. İftar: Decentralized supervisory controller design to avoid deadlock in Petri nets. Internat. J. Control *76* (2003), 1285–1295. Also see: A. Aybar and A. İftar: Corrections to decentralized supervisory controller design to avoid deadlock in Petri nets. Internat. J. Control *76* (2003), p. 1584.

[6] A. Aybar, A. İftar, and H. Apaydın-Özkan: Centralized and decentralized supervisory controller design to enforce boundedness, liveness, and reversibility in Petri nets. Internat. J. Control *78* (2005), 537–553.

[7] L. Bakule, J. Rodellar, and J. M. Rossell: Generalized selection of complementary matrices in the inclusion principle. IEEE Trans. Automat. Control *45* (2000), 1237–1243.

[8] K. Barkaoui and I. B. Abdallah: Deadlock avoidance in FMS based on structural theory of Petri nets. In: Proc. 1995 IEEE Symposium on the Emerging Technology and Factory Automation, Paris 1995, pp. 499–510.

[9] C. G. Cassandras and S. Lafortune: Introduction to Discrete Event Systems. Kluwer Academic Publishers, Norwell 1999.

[10] R. Cordone, L. Ferrarini, and L. Piroddi: Some results on the computation of minimal siphons in Petri nets. In: Proc. IEEE Conference on Decision and Control, Maui 2003, pp. 3754–3759.

[11] J. Esparza and M. Nielsen: Decidability Issues in Petri Nets. BRICS Report Series, RS-94-8, Department of Computer Science, University of Aarhus 1994.

[12] M. P. Fanti, B. Maione, and B. Turchiano: Comparing digraph and Petri net approaches to deadlock avoidance in FMS. IEEE Trans. Systems, Man and Cybernetics, Part B *30* (2000), 783–798.

[13] M. P. Fanti and M. Zhou: Deadlock control methods in automated manufacturing systems. IEEE Trans. Systems, Man, and Cybernetics, Part A *34* (2004), 5–22.

[14] Y. Ho (ed.): Discrete Event Dynamic Systems: Analyzing Complexity and Performance in the Modern World. A Selected Reprint Volume. The Institute of Electrical and Electronics Engineers, New York 1992.

[15] L. E. Holloway and B. H. Krogh: On closed-loop liveness of discrete-event systems under maximally permissive control. IEEE Trans. Automat. Control *37* (1992), 692–697.

[16] A. İftar: Overlapping decentralized dynamic optimal control. Internat. J. Control *58* (1993), 187–209.

[17] A. İftar: Decentralized estimation and control with overlapping input, state, and output decomposition. Automatica *29* (1993), 511–516.

[18] A. İftar and Ü. Özgüner: Local LQG/LTR controller design for decentralized systems. IEEE Trans. Automat. Control *32* (1987), 926–930.

[19] A. İftar and Ü. Özgüner: Contractible controller design and optimal control with state and input inclusion. Automatica *26* (1990), 593–597.

[20] A. İftar and Ü. Özgüner: Overlapping decompositions, expansions, contractions, and stability of hybrid systems. IEEE Trans. Automat. Control *43* (1998), 1040–1055.

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

[22] M. Ikeda and D. D. Šiljak: Overlapping decentralized control with input, state, and output inclusion. Control Theory Adv. Technol. *2* (1986), 155–172.

[23] M. Ikeda, D. D. Šiljak, and D. E. White: Decentralized control with overlapping information sets. J. Optim. Theory Appl. *34* (1981), 279–310.

[24] M. V. Iordache and P. J. Antsaklis: Software Tools for the Supervisory Control of Petri Nets Based on Place Invariants. Tech. Rep. ISIS-2002-003, the ISIS Group at the University of Notre Dame, 2002.

[25] M. V. Iordache and P. J. Antsaklis: Design of $\mathcal{T}$-liveness enforcing supervisors in Petri nets. IEEE Trans. Automat. Control *48* (2003), 1962–1974.

[26] M. V. Iordache, J. O. Moody, and P. J. Antsaklis: A method for the synthesis of liveness enforcing supervisors in Petri nets. In: Proc. American Control Conference, Arligton 2001, pp. 4943–4948.

[27] Z. W. Li and M. C. Zhou: Elementary siphons of Petri nets and their application to deadlock prevention in flexible manufacturing systems. IEEE Trans. Systems, Man and Cybernetics, Part A, *34* (2004), 38–51.

[28] Y. Ohta, D. D. Šiljak, and T. Matsumoto: Decentralized control using quasi-block diagonal dominance of transfer function matrices. IEEE Trans. Automat. Control *31* (1986), 420–430.

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

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

[31] M. Uzam and M. Zhou: An iterative synthesis approach to Petri net-based deadlock prevention policy for flexible manufacturing systems. IEEE Trans. Systems, Man, and Cybernetics, Part A, *37* (2007), 362–371.

[32] A. R. Wang, Z. W. Li, and J. Y. Jia: A Petri nets based deadlock avoidance policy for flexible manufacturing systems. In: Proc. 30th Annual Conference of the IEEE Industrial Electronics Society, Busan 2004, pp. 1896–1901.

[33] N. Viswanadham, Y. Narahari, and T. L. Johnson: Deadlock prevention and deadlock avoidance in flexible manufacturing systems using Petri net models. IEEE Trans. Robotics and Automation *6* (1990), 713–723.

[34] M. Zhou and F. Di Cesare: Petri Net Synthesis for Discrete Event Control of Manufacturing Systems. Kluwer Academic Publishers, Norwell 1993.

Aydın Aybar and Altuğ İftar, Anadolu University, Department of Electrical and Electronics Engineering, 26555, Eskişehir. Turkey.
e-mail: aaybar@anadolu.edu.tr and aiftar@anadolu.edu.tr