Int. J. of Computers, Communications & Control, ISSN 1841-9836, E-ISSN 1841-9844
Vol. V (2010), No. 4, pp. 447-457

Decentralized Controller Design for Forbidden States Avoidance in Timed Discrete Event Systems

A. Aybar

Aydın Aybar
Anadolu University, Dept. of Electrical and Electronics Engineering
26555, Eskişehir, Turkey. E-mail: aaybar@anadolu.edu.tr

Abstract: A decentralized controller design approach is developed for the timed discrete event systems which are modelled by timed automata in this work. An approach, called *augmentation*, is presented to obtain the new modelling method such that each unit delay of any event represents a pair of new state and event. The augmented automata model, obtained by using this approach, is considered to design a decentralized controller. This controller design approach is developed such that the local controller is designed for each subautomaton, obtained by using overlapping decompositions and expansions and these controllers are then combined to obtain a decentralized controller for the given timed automaton. The designed decentralized controller guarantees the unreachability of a forbidden state in the considered automaton.

Keywords: Discrete event systems, Automata, Time delays, Decentralized controller.

1 Introduction

Although automata and Petri nets are known as common modelling methods for discrete event systems (see, [1]– [4]), these models were first presented without time notation. Since there exist time delays in the dynamic systems, time notation is a necessity for the modelling methods of the discrete event systems [5,6]. Time notation was used for automata (see, [7]). In this timed automata model, a class of finite state automata was extended with a set of clocks. The clocks were chosen as real values and timed event, denoted by a pair of an event and its occurence time, were used to determine the reachability of any state. Afterwards, the timed automata model was used by many works (for example, [8–10]). Moreover, the basic supervisory controller approaches were presented for these timed systems, modelled by timed automata (for example, [11–13]).

It is known that the computational complexity of a supervisory controller design depends on the number of states and clocks for the timed automata model [10]. Moreover, the computational complexity increases exponentially with the number of states of the untimed automata model [14] (also see, [15]). Thus, a controller design for timed automata (especially, large scale automata have more number of states and events), can be more complex. An approach, called augmentation, is first introduced for timed automata in order to decrease the computational complexity, depending on the time and/or clock, of a controller design.

This approach, based on [16, 17], is described such that each unit delay of any event represents a pair of new state and event, and then these pairs are added to the original automaton. A new modelling model is introduced such that the augmented automaton is obtained by adding the pairs of events and states, corresponding to unit time delays, to the original automaton in this work. In [16, 17], the streching approach was developed for timed Petri nets. In this developed approach, each delay, assigned to a transition, denotes a pair of new place and transition. Using the similarity between automata and Petri nets, we first develop the augmentation approach in this work. Although any event of automata can be related to any transition of Petri net, there exist some differences between these models (for example, a marking vector of Petri net is corresponding to a state of automaton).

The augmented automaton is used to design a decentralized controller in this work. An algebraic approach, which gives the state space representation for automata, is also developed to determine the state vectors.

Our aim is a decentralized controller design which prevents the occurence of the forbidden states. To facilitate the controller design, we use the approach of overlapping decompositions. The overlapping decompositions approach was first introduced by [18] for the case of continuous-state systems (systems described by differential or difference equations with continuous state variables). This approach was then used for discrete event systems by ([4, 14, 19, 20]).

2 Preliminaries

2.1 Mathematical Model

The timed automata model is represented by $\mathcal{A}(Q,\Sigma,\mathcal{C},q_0,D)$. Here, Q is the set of states, Σ is the set of events, $\mathcal{C}: Q\times Q \to \Sigma\cup\{0\}$ is the connection matrix, q_0 is the initial state at the initial time, and D is set of the time delays of the events such that $d_e \in \mathcal{R}^+$ is the time delay of the event $e\in\Sigma$, where $\mathcal{R}^+$ is set of nonnegative real numbers.

The connection matrix is given as

$$\mathcal{C}(q_i,q_j) := \begin{cases} e, & \text{if } q_i \text{ is obtained when event } e \text{ occurs at state } q_j \\ 0, & \text{otherwise} \end{cases}, \text{ for } q_i,q_j\in Q$$

$\mathcal{C}(q_i,q_j)=0$ denotes no connection between two states q_i and q_j. $\mathcal{C}(q_i,q_j)=e$ denotes a connection between these states via e. It is assumed that the connection of between two states is done by only one event.

In this work, the vector-matrix form is used to determine new state. The state vector at time τ is denoted by $S(\tau)$

$$S(\tau) := \begin{cases} \Lambda_Q(q), & \text{if } \tau=\Gamma(q), \text{ for any } q\in Q \\ \mathcal{Z}, & \text{otherwise} \end{cases}$$

Here, $\Gamma(q)$ denotes the obtained time of state q (it is assumed that each event occurs immediately as it becames possible), $\Lambda_Q: Q\to\{0,1\}^{|Q|}$, $\Lambda_Q(q) := \begin{cases} 1, & \text{if } q=[Q]_j \\ 0, & \text{otherwise} \end{cases}$, $j\in\{1,...,|Q|\}$ where, $[Q]_j$ denotes the jth element of Q, and $|Q|$ indicates the number of the elements of Q, τ denotes the global time, Z, which is zeros vector, denotes that the occurence of the event e has not finished at τ or the considered event can not occured at the given state.

The state equation is given as follows:

$$S(\tau) = (\mathcal{C}\vee S(\tau_e))\wedge\mathcal{O}(e,\tau_e), \;\; e\in\Sigma \tag{1}$$

It is assumed that the initial state $S(\tau_0)=\Lambda_{\bar{Q}}(q_0)$ and there exists an event e such that $\tau_e=\Gamma(q)$ for $q\in Q$ (there is one exception such that if there exists deadlock in the considered automaton, no event can occur at deadlock state), where τ_e denotes the occurence time of the event e. Note that, $\vee$ and $\wedge$ are used respectively. Here,

- The event function is defined as $\mathcal{O}(e,\tau_e) := e\odot\phi(\tau-\tau_e-d_e)$, where, $\phi:\mathcal{R}^+\to\{0,1\}$; $\phi(x)=\begin{cases} 1, & \text{if } x\geq 0 \\ 0, & \text{otherwise} \end{cases}$.
- The operation $\odot$ on the set $\ddot{\Sigma} := \Sigma\cup\{0,1\}$ is defined as $0\odot 0=0,\; e_k\odot 0=0,\; 0\odot e_k=0,\; 0\odot 1=0,$ $1\odot 0=0,\; 1\odot 1=0,\; e_k\odot 1=e_k,\; 1\odot e_k=e_k,\; e_k\odot e_k=0,\; e_k\odot e_l=0,\; e_l\odot e_k=0.$

- The operation $\vee$ on the matrices $H \in \ddot{\Sigma}^{|Q|\times|Q|}$ and $R \in \ddot{\Sigma}^{|Q|}$, $F = H \vee R$, is defined as $F(i) = \sum_{k=1}^{|Q|} H(i,k) \odot R(k)$, $i \in \{1,...,|Q|\}$.

- The operation $\otimes$ on the set $\ddot{\Sigma}$ is defined as $0 \otimes 0 = 0$, $e_i \otimes 0 = 0$, $0 \otimes e_i = 0$, $0 \otimes 1 = 0$, $1 \otimes 0 = 0$, $1 \otimes 1 = 1$, $e_i \otimes 1 = e_i$, $1 \otimes e_i = e_i$, $e_i \otimes e_i = 1$, $e_i \otimes e_j = 0$, $e_j \otimes e_i = 0$.

- For $F \in \ddot{\Sigma}^{|Q|}$ and $e \in \Sigma$, $F \wedge e = [F(1) \otimes e \; ... \; F(|Q|) \otimes e]^T$.

In the given example automaton, shown in Fig. 1a, the set of states is $Q = \{q_0, q_1, q_2, q_3, q_4\}$, the set of events is $\Sigma = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7\}$, and the initial state is q_0. The set of time delays, assigned to the events, is given as $d_{e_1} = d_{e_2} = d_{e_6} = 2$ sec., $d_{e_3} = 3$ sec., $d_{e_4} = d_{e_5} = d_{e_7} = 1$ sec. Let the occurence time of e_3 be $\tau_{e_3} = 5$ sec. and $S(5) = \Lambda_Q(q_2) = [0\ 1\ 0\ 0\ 0]^T$. The state vector is obtained as

$$\begin{aligned}
S(\tau) &= (\mathcal{C} \vee S(5)) \wedge \mathcal{O}(e_3, 5) = \left(\begin{bmatrix} 0 & 0 & 0 & e_4 & 0 \\ e_1 & 0 & 0 & 0 & 0 \\ 0 & e_3 & 0 & e_5 & 0 \\ 0 & e_2 & 0 & 0 & e_6 \\ e_7 & 0 & 0 & 0 & 0 \end{bmatrix} \vee \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \right) \wedge (e_3 \odot \phi(\tau - 5 - 3)) \\
&= \begin{bmatrix} 0 \\ 0 \\ e_3 \odot 1 \\ 0 \\ 0 \end{bmatrix} \wedge (e_3 \odot \phi(\tau - 5 - 3)) = \begin{cases} [0\,0\,1\,0\,0]^T, & \text{if } \tau \geq 8 \\ [0\,0\,0\,0\,0]^T, & \text{if } 5 < \tau < 8 \end{cases}
\end{aligned}$$

q_2 is obtained when the occurence of event e_3 is completed (q_2 is not yet obtained in time interval between 5 sec. and 8 sec.). In this work, a new model is introduced in next section and used to design a decentralized controller.

2.2 Overlapping Decompositions and Expansions

Overlapping decompositions and expansions [21] have been widely used to design decentralized controllers for continuous-state systems. These concepts have also been used to design supervisory controllers for discrete event systems modeled by Petri nets [19] and by automata [14]. To our best knowledge, overlapping decompositions and expansions of discrete event systems modelled by automata or formal languages have been first introduced in [14]. In the given approach, *overlapping subautomata* of an automaton are first identified by examining the topological structure of the given automaton. These subautomata are identified such that the only interconnection between the subautomata are through the overlapping part, i.e., no event should connect two states in different subautomata, unless one of these states is in the overlapping part of the two subautomata. As an example, the automaton (Fig. 1a) can be decomposed into two subautomata as shown in Fig. 1b-1c ([14]).

After an overlapping decomposition of the original automaton is obtained, the expansions of the automaton is explained as follows [14]:

i) A state or an event in the overlapping part of n subautomata is repeated n times and each repeated state/event is assigned to a different subautomaton.

ii) Two events are introduced between any two repeated states, such that when such an event occurs the state changes from one repeated state to the other. Note that, a delay of each new event is assigned to the biggest common divisor of time delays of the original automaton in this work.

iii) If the initial state is in the overlapping part of the original automaton, then the initial state of the expanded automaton can be chosen as any one of the repeated states of the original initial state.

Otherwise, the initial state of the expanded automaton is chosen as the initial state of the original automaton.

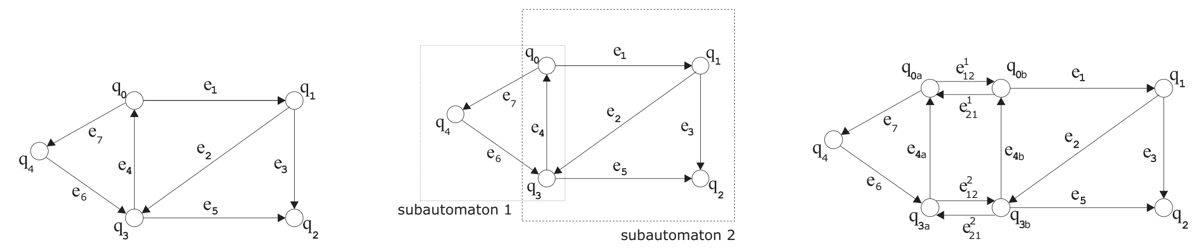

Figure 1. (a) Example automaton (b) Overlappingly decomposed automaton (c) Expanded automaton

As a result of this procedure, an expanded automaton, which consists of α disjoint subautomata, is obtained from an original automaton which was decomposed into α overlapping subautomata.

The set of states of the expanded automaton is given by $\tilde{Q} := \cup_{i=1}^{\alpha} Q_i$, where Q_i is the set of states of the i^{th} subautomaton. The set of events of the expanded automaton is given by $\tilde{\Sigma} := \breve{\Sigma} \cup \grave{\Sigma}$. Here, $\breve{\Sigma} = \cup_{i=1}^{\alpha} \Sigma_i$, where Σ_i is the set of events of the i^{th} subautomaton and $\grave{\Sigma}$ is the set of additional events introduced between the repeated states. As an example, the states q_0, q_3, and the event e_4 are repeated, and new events e^1_{12}, e^1_{21}, e^2_{12} and e^2_{21} are added to the repeated states in the expanded automaton in Fig. 1a. Then, the time delay of these events is determined as one second.

3 New Model for Timed Automata

Although the usage of time in the mathematical model is a necessity for the real world system, the computational complexity of time delay systems increases because of the defined all processes and functions need more memories and time. We introduce the augmentation approach. Using this approach, a new model is obtained for timed automata and called as the augmented automata, where each event has only unit time delay.

The augmentation approach is defined such that time delays are represented by new states and events in this work. The augmented automaton, $\bar{A}_T(\bar{Q},\bar{\Sigma},\bar{\mathcal{C}},q_0)$, is introduced, where, $\bar{\mathcal{C}} : \bar{Q} \times \bar{Q} \to \bar{\Sigma}$, $\bar{Q} := Q \cup (\bigcup\limits_{e\in\Sigma} \Delta_S(e))$, and $\bar{\Sigma} := \Sigma \cup (\bigcup\limits_{e\in\Sigma} \Delta_E(e))$ are given following items.

- The time delays of the events are scaled such that $d^s_e := d_e/\lambda$, for $e \in \Sigma$ and $d_e \in D$, where λ indicates the biggest common divisor of time delays. Note that, the set of the scaled time delays of the events is denoted by D_s. It is assumed that $d^s_e \geq 1$, for all $e \in \Sigma$ in this work.

- For the event $e \in \Sigma$ such that $\mathcal{C}(q_i,q_j) = e$ and $d^s_e = 1$, the input connection from e to the state is hold such as $\bar{\mathcal{C}}(q_i,q_j) = \mathcal{C}(q_i,q_j) = e$ for $q_i,q_j \in Q$. Note that, if $\mathcal{C}(q_a,q_b) = 0$, then $\bar{\mathcal{C}}(q_a,q_b) = 0$.

- For the event $e^* \in \Sigma$, and $d^s_{e^*} > 1$, $\delta_{e^*} := d^s_{e^*} - 1$ numbers new events and states are defined such as $f^{e^*}_1$, $f^{e^*}_2$, ... , $f^{e^*}_{\delta_{e^*}}$, and $p^{e^*}_1$, $p^{e^*}_2$, ... , $p^{e^*}_{\delta_{e^*}}$. The sets are constructed by using these events and states as $\Delta_E(e^*)$ and $\Delta_S(e^*)$, respectively.

- The pairs are constructed by using the new events and states for any event $e^* \in \Sigma$, $d^s_{e^*} > 1$, such that $(f^{e^*}_i, p^{e^*}_i)$ for $i \in \{1,2,...,\delta_e\}$. For $\bar{\mathcal{C}}(q_k,q_n) = e^*$, the connections are described such as from q_n to $f^{e^*}_1$, from $f^{e^*}_1$ to $p^{e^*}_1$, from $p^{e^*}_1$ to $f^{e^*}_2$, ... from $f^{e^*}_{\delta_{e^*}}$ to q_k. Hence, the new connection matrix is constructed for the new automaton model such as $\bar{\mathcal{C}}(p^{e^*}_1, q_n) = f^{e^*}_1$, $\bar{\mathcal{C}}(p^{e^*}_2, p^{e^*}_1) = f^{e^*}_2$, ..., $\bar{\mathcal{C}}(q_k, p^{e^*}_{\delta_{e^*}}) = e^{e^*}_{\delta_{e^*}}$.

As a result of the above procedure, we obtain the augmented automaton which has more events and states but each event has only unit time delay.

We introduce the algebraic approach for the augmented automaton. Let $\bar{S}_n$ be denote the present state vector and $\bar{S}_{n+1}$ be denote the next state vector (for $n \in \{0,1,2,...\}$, $\bar{S}_0 = \Lambda_{\bar{Q}}(q_0)$ denotes the initial state vector). The state equation is defined as follows:

$$\bar{S}_{n+1} = (\bar{\mathcal{C}} \vee \bar{S}_n) \wedge \bar{e}, \ \ \bar{e} \in \bar{\Sigma}. \tag{2}$$

It is possible to obtain p_k^e as the current state. It shows that the occurence of the event e has not finished yet and also the duration time is determined as $k * \lambda + \tau_e$ for the event e. Compared to (1), the evaluation of the above equation (2) is much simpler, since it does not require the time notation and the event function $\mathcal{O}$.

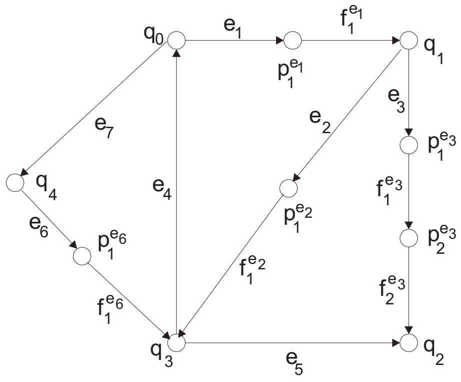

Figure 2. Augmented automaton

For example, we obtain the augmented automaton (Fig. 2.) for the given timed automata (Fig. 1a). The set of states is $\bar{Q} = \{q_0,q_1,q_2,q_3,q_4\} \cup \{p_1^{e_1}, p_1^{e_2}, p_1^{e_3}, p_2^{e_3}, p_1^{e_6}\}$, where, $\Delta_S(e_1) = \{p_1^{e_1}\}$, $\Delta_S(e_2) = \{p_1^{e_2}\}$, $\Delta_S(e_3) = \{p_1^{e_3}, p_2^{e_3}\}$, and $\Delta_S(e_6) = \{p_1^{e_6}\}$, the set of events is $\bar{\Sigma} = \{e_1,e_2,e_3,e_4,e_5,e_6\} \cup \{f_1^{e_1}, f_1^{e_2}, f_1^{e_3}, f_2^{e_3}, f_1^{e_6}\}$, where $\Delta_E(e_1) = \{f_1^{e_1}\}$, $\Delta_E(e_2) = \{f_1^{e_2}\}$, $\Delta_E(e_3) = \{f_1^{e_3}, f_2^{e_3}\}$, and $\Delta_E(e_6) = \{f_1^{e_6}\}$, and the connection matrix is given as

$$\bar{\mathcal{C}} = \begin{bmatrix} 0 & 0 & 0 & e_4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & f_1^{e_1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & e_5 & 0 & 0 & 0 & 0 & f_2^{e_3} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & f_1^{e_2} & 0 & 0 & f_1^{e_6} \\ e_7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ e_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & e_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & e_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & f_1^{e_3} & 0 & 0 \\ 0 & 0 & 0 & 0 & e_6 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

4 Decentralized Controller Design

A decentralized controller for the forbidden states avoidance is developed for the considered automaton in this section.

4.1 Centralized Control

The centralized controller guarantees the unreachability of a forbidden state for the original automaton ($\mathcal{F}$ denotes the set of the forbidden states). Now, we consider a centralized controller design for the original augmented automaton (OAA).

In the OAA, the set of forbidden states is taken as $\bar{\mathfrak{F}} = \mathfrak{F}$. It is possible that there exists a state which only leads to any element of the set $\bar{\mathfrak{F}}$. Thus, the set $\bar{\mathfrak{F}}$ is extented by these sets and a new set, denoted by $\bar{\mathfrak{G}}$, is obtained by the following algorithm. This algorithm, called as FS, requests the set of the forbidden states and the definition. Note that, this algorithm also finds the deadlock states, in which no event can occur, and adds these deadlock states to the set $\bar{\mathfrak{G}}$. In this work, each element of $\bar{\mathfrak{G}}$ is called as forbidden state.

A controller for the OAA is defined as

$$\bar{K}(\bar{S}_n, \bar{e}) = \bar{K}(\Lambda_{\bar{Q}}(q), \bar{e}) = \begin{cases} 0, & \text{if } \bar{S}_{n+1} \in \bar{\mathfrak{G}}_v \\ 1, & \text{otherwise} \end{cases}, \ \bar{e} \in \bar{\Sigma} \tag{3}$$

where, $\bar{S}_n = \Lambda_{\bar{Q}}(q)$, $\bar{S}_{n+1} = (\bar{\mathfrak{C}} \vee \bar{S}_n) \wedge \bar{e}$ and $\bar{\mathfrak{G}}_v := \bigcup_{q \in \bar{\mathfrak{G}}} \Lambda_{\bar{Q}}(q)$ denotes the set of the state vectors, corresponding to states of $\bar{\mathfrak{G}}$. Note that, if q_f is a forbidden state, $q_f \in \bar{\mathfrak{G}}$, then $\Lambda_{\bar{Q}}(q_f)$ is called as forbidden state vector, $\Lambda_{\bar{Q}}(q_f) \in \bar{\mathfrak{G}}_v$. Once $\bar{K}(\bar{S}_n, \bar{e}) = 0$ denotes disabling event $\bar{e} \in \bar{\Sigma}$, $\bar{K}(\bar{S}_n, \bar{e}) = 1$ denotes enabling event $\bar{e}$. Then, this controller guarantees the unreachability of an element of $\bar{\mathfrak{G}}$.

The OAA with the controller can be also called as controlled automaton, denoted by $\bar{\mathcal{A}}_T^K(\bar{Q}, \bar{\Sigma}, \bar{\mathfrak{C}}, q_0, \bar{K})$. The controlled state equation, which is obtained by adding this controller to the equation (2), is given as follows:

$$\bar{S}_{n+1} = (\bar{\mathfrak{C}} \vee \bar{S}_n) \wedge (\bar{e} \otimes \bar{K}(\bar{S}_n, \bar{e})), \ \bar{e} \in \bar{\Sigma} \tag{4}$$

Thus, any element of of $\bar{\mathfrak{G}}$ does not occur in this controlled automaton.

Algorithm to construct the set $\bar{\mathfrak{G}}$

```
$\bar{\mathfrak{G}}$ = FS($\bar{A}_T, \bar{\mathfrak{F}}$)
$\bar{\mathfrak{G}} = \bar{\mathfrak{F}}$
Do Loop Construction
  $\hat{\mathfrak{F}} = \emptyset$
  For $i = 1$ to $|\bar{Q}|$
    If $[\bar{Q}]_i \notin \bar{\mathfrak{G}}$ Then
      $cnt = 0$
      For $j = 1$ to $|\bar{Q}|$
        If $\bar{C}(j,i) = 0$ Or $[\bar{Q}]_j \in \bar{\mathfrak{G}}$ Then
          $cnt = cnt + 1$
          If $cnt = |\bar{Q}|$ Then
            $\hat{\mathfrak{F}} \leftarrow \hat{\mathfrak{F}} \,\hat{\cup}\, \{[\bar{Q}]_j\}$
          End
        End
      End
    End
  End
  If $\hat{\mathfrak{F}} = \emptyset$ Then
    Exit Loop Construction
  End
  $\bar{\mathfrak{G}} \leftarrow \bar{\mathfrak{G}} \,\hat{\cup}\, \hat{\mathfrak{F}}$
Loop Construction
```

Here, both $\cup$ and $\hat{\cup}$ are used to denote the *set union*. $L \,\hat{\cup}\, M$ is used, rather than $L \cup M$, whenever it is known apriori that $L \cap M = \emptyset$. To evaluate $N = L \cup M$, the set C is first initialized as L; for each element (from first to last), m, of M, it is then checked whether $m \in L$. If $m \notin L$, then m is added to set N. To evaluate $N = L \,\hat{\cup}\, M$, on the other hand, elements of L and of M are simply appended to form N.

4.2 Decentralized Control

Now, we first consider to design a controller for each disjoint subautomaton. Then, a controller of the expanded augmented automaton (EAA) is obtained by using these controllers of subautomata. Finally, a decentralized controller is designed by using the controller of the EAA for the OAA.

It is known that the augmented subautomata are easily obtained by using overlapping decompositions and expansions. Let $\bar{A}_{i_T}(\bar{Q}_i, \bar{\Sigma}_i, \bar{\mathcal{C}}_i, q_{i_0})$ be denote the ith subautomaton. Now, some definitions and notation are given such that the EAA is denoted by $\tilde{A}_T(\tilde{Q}, \tilde{\Sigma}, \tilde{\mathcal{C}}, \tilde{q}_0)$, where, $\tilde{Q} := \cup_{i=1}^{\alpha} \bar{Q}_i$, $\tilde{\Sigma} := \cup_{i=1}^{\alpha} \bar{\Sigma}_i \cup \grave{\Sigma}$, and the connection matrix can be easily determined by using new sets of states and events.

$\Psi_{\bar{Q}} : \bar{Q} \to \tilde{Q}$, $\Psi_{\bar{Q}}(q)$ denotes the set of states in the EAA which corresponds to the state q in the OAA and $\Psi_{\bar{\Sigma}} : \bar{\Sigma} \to \tilde{\Sigma}$, $\Psi_{\bar{\Sigma}}(e)$ denotes the set of events in the EAA which corresponds to the event e in the OAA. Also, we define $\Psi_{\bar{\Sigma}}^{-1} : \tilde{\Sigma} \to \bar{\Sigma}$ and $\Psi_{\bar{Q}}^{-1} : \tilde{Q} \to \bar{Q}$ such that $e = \Psi_{\bar{\Sigma}}^{-1}(\tilde{e}) \iff \tilde{e} \in \Psi_{\bar{\Sigma}}(e)$ and $q = \Psi_{\bar{Q}}^{-1}(\tilde{q}) \iff \tilde{q} \in \Psi_{\bar{Q}}(q)$.

The set of the forbidden states for the ith augmented subautomaton is obtained as $\bar{\mathfrak{F}}_i := \tilde{\mathfrak{F}} \cap \bar{Q}_i$, where $\tilde{\mathfrak{F}} = \bigcup_{\bar{q} \in \bar{\mathfrak{F}}} \Psi_{\bar{Q}}(\bar{q})$. For the ith subautomaton, $\bar{\mathfrak{G}}_i$ and $\bar{\mathfrak{G}}_{i_v}$ are obtained by using the algorithm *FS*. Note that, this algorithm needs the definition of the ith subautomaton, and the set $\bar{\mathfrak{F}}_i$.

It is possible to design a controller, $\bar{K}_i$ for $\bar{A}_{i_T}$, if the initial state of this subautomaton is not a forbidden state ($q_{i_0} \notin \bar{\mathfrak{G}}_i$). Since this repeated state is used for the interconnection between the subautomata (see, Section 2.2), it is assumed that any repeated state in the ith subautomaton is not element of $\bar{\mathfrak{G}}_i$ for all $i \in \{1, ..., \alpha\}$ ($\tilde{q} \notin \bar{\mathfrak{G}}_i$ for $\tilde{q} \in \tilde{Q}_i^o$ which denotes the set of repeated states in the ith subautomaton).

The controller for the EAA is designed by using local controllers, $\bar{K}_i$ for all $i \in \{1, ..., \alpha\}$, where α denotes the number of subautomata,

$$\tilde{K}(\Lambda_{\tilde{Q}}(\tilde{q}), \tilde{e}) = \begin{cases} \bar{K}_i(\Lambda_{\bar{Q}_i}(\tilde{q}), \tilde{e}), & \text{if } \tilde{e} \in \bar{\Sigma}_i \\ 1, & \text{otherwise} \end{cases}, \quad \tilde{q} \in \tilde{Q} \tag{5}$$

Note that, $\tilde{\mathfrak{G}} := \bigcup_{i \in \{1,...,\alpha\}} \bar{\mathfrak{G}}_i$. Consequently, the controlled state equation is obtained by adding this controller to the state equation, for $\tilde{S}_n = \Lambda_{\tilde{Q}}(\tilde{q})$,

$$\tilde{S}_{n+1} = (\tilde{\mathcal{C}} \vee \tilde{S}_n) \wedge (\tilde{e} \otimes \tilde{K}(\tilde{S}_n, \tilde{e})), \; \tilde{e} \in \tilde{\Sigma}$$

Theorem 1: $\tilde{K}$ avoids the existence of the elements of $\tilde{\mathfrak{G}}$ in $\tilde{A}_T^K$.
Proof: Let $\tilde{q} \in \tilde{Q}$ and $\tilde{e} \in \tilde{\Sigma}$. This state is also an element of any subautomaton, $\tilde{q} \in \bar{Q}_k$, for $k \in \{1, ..., \alpha\}$.

i) If there is no relation between $\tilde{q}$ and $\tilde{e}$, then $\tilde{K}(\Lambda_{\tilde{Q}}(\tilde{q}), \tilde{e}) = 1$ because of its definition (see, the equation (5)). In this case, $\tilde{e}$ is not occured at $\tilde{q}$ and also this value of $\tilde{K}$ does not affect the controlled state equation because of the definition of operation $\otimes$.

ii) If $\tilde{e} \in \grave{\Sigma}$, then $\tilde{K}(\Lambda_{\tilde{Q}}(\tilde{q}), \tilde{e}) = 1$. In this case, the next state is also repeated state and not a forbidden state ($\tilde{q}^o \notin \bar{\mathfrak{G}}_j$ for $\tilde{q}^o \in \tilde{Q}_j^o$, $\forall j \in \{1, ..., \alpha\}$).

iii) If $\tilde{e} \in \bar{\Sigma}_k$, then $\tilde{K}(\Lambda_{\tilde{Q}}(\tilde{q}), \tilde{e}) = \bar{K}_k(\Lambda_{\bar{Q}_k}(\tilde{q}), \tilde{e})$. Let the next state, $\tilde{q}^+$, be obtained by using $\tilde{e}$ from $\tilde{q}$. If $\tilde{q}^+ \in \bar{\mathfrak{G}}_k$, then $\bar{K}_k(\Lambda_{\bar{Q}_k}(\tilde{q}), \tilde{e}) = 0$ and $\tilde{q}^+ \in \tilde{\mathfrak{G}}$ because of definition of $\tilde{\mathfrak{G}}$ $[\tilde{K}(\Lambda_{\tilde{Q}}(\tilde{q}), \tilde{e}) = 0]$. Otherwise, $\bar{K}_k(\Lambda_{\bar{Q}_k}(\tilde{q}), \tilde{e}) = 1$ and $\tilde{q}^+ \notin \tilde{\mathfrak{G}}$ $[\tilde{K}(\Lambda_{\tilde{Q}}(\tilde{q}), \tilde{e}) = 1]$. Note that, $\tilde{K}(\Lambda_{\tilde{Q}}(\tilde{q}), \tilde{e}) = 1$ if there is no relation between $\tilde{q}$ and $\tilde{e}$. $\square$

We now obtain a controller, $\bar{K}$, for the OAA, by using the controller, $\tilde{K}$, for the EAA as follows:

$$\bar{K}(\Lambda_{\bar{Q}}(\bar{q}), \bar{e}) = \prod_{\tilde{q} \in \Psi_{\bar{Q}}(\bar{q})} \prod_{\tilde{e} \in \Psi_{\bar{\Sigma}}(\bar{e})} \tilde{K}(\tilde{\Lambda}(\tilde{q}), \tilde{e}), \; \bar{q} \in \bar{Q}, \; \bar{e} \in \bar{\Sigma} \tag{6}$$

Furthermore, the controlled state equation is given as $\bar{S}_{n+1} = (\bar{\mathcal{C}} \vee \bar{S}_n) \wedge (\bar{e} \otimes \bar{K}(\bar{S}_n, \bar{e}))$, $\bar{e} \in \bar{\Sigma}$, where, $\bar{S}_n = \Lambda_{\bar{Q}}(\bar{q})$, in the OAA.

Theorem 2: $\bar{K}$ guarantees the unreachability of a forbidden state in $\bar{\mathcal{A}}_T^K$.

Proof: It is known that the sets $\bar{\mathcal{G}}_j$ for all $j \in \{1,...,\alpha\}$ and $\tilde{\mathcal{G}}$ are determined. Any repeated event in the EAA is only connected to the repeated states because of overlapping decomposition approach.

i) if $\bar{q}^{\ddagger} \in \bar{Q}$, $\Psi_{\bar{Q}}(\bar{q}^{\ddagger}) = \{\bar{q}^{\ddagger}\}$ and $\bar{e}^* \in \bar{\Sigma}$, $\Psi_{\bar{\Sigma}}(\bar{e}^*) = \{\bar{e}_a^*, \bar{e}_b^*, ..., \bar{e}_x^*\}$, then
$\bar{K}(\Lambda_{\bar{Q}}(\bar{q}^{\ddagger}), \bar{e}^*) = \tilde{K}(\Lambda_{\bar{Q}}(\bar{q}^{\ddagger}), \bar{e}_a^*)$.
$\tilde{K}(\Lambda_{\bar{Q}}(\bar{q}^{\ddagger}), \bar{e}_b^*)$. ... $\tilde{K}(\Lambda_{\bar{Q}}(\bar{q}^{\ddagger}), \bar{e}_x^*)$. Since any event, in the overlapping part, is only connected to the states in the overlapping part, $\bar{K}(\Lambda_{\bar{Q}}(\bar{q}^{\ddagger}), \bar{e}^*) = 1$.

ii) if $\bar{q}^{\ddagger} \in \bar{Q}$, $\Psi_{\bar{Q}}(\bar{q}^{\ddagger}) = \{\bar{q}^{\ddagger}\}$ and $\bar{e}^* \in \bar{\Sigma}$, $\Psi_{\bar{\Sigma}}(\bar{e}^*) = \{\bar{e}^*\}$, then $\bar{q}^{\ddagger}$ and $\bar{e}^*$ are elements of subautomata. Note that, $\bar{K}(\Lambda_{\bar{Q}}(\bar{q}^{\ddagger}), \bar{e}^*) = \tilde{K}(\Lambda_{\bar{Q}}(\bar{q}^{\ddagger}), \bar{e}^*) = 0$ if $\bar{q}^{\ddagger}$ and $\bar{e}^*$ are not in same automaton. If there is a relation between $\bar{q}^{\ddagger}$ and $\bar{e}^*$ in the jth subautomaton, then $\bar{K}(\Lambda_{\bar{Q}}(\bar{q}^{\ddagger}), \bar{e}^*) = \tilde{K}(\Lambda_{\bar{Q}}(\bar{q}^{\ddagger}), \bar{e}^*) = \bar{K}_j(\Lambda_{\bar{Q}_j}(\bar{q}^{\ddagger}), \bar{e}^*)$. In this case, if $\bar{q}^u$, which is obtained by using $\bar{e}^*$ from $\bar{q}^{\ddagger}$, is an element of $\bar{\mathcal{G}}_j$, then $\bar{K}_j(\Lambda_{\bar{Q}_j}(\bar{q}^{\ddagger}), \bar{e}^*) = 0$. Otherwise, $\bar{K}_j(\Lambda_{\bar{Q}_j}(\bar{q}^{\ddagger}), \bar{e}^*) = 1$.

iii) If $\bar{q}^{\ddagger} \in \bar{Q}$, $\Psi_{\bar{Q}}(\bar{q}^{\ddagger}) = \{\bar{q}_a^{\ddagger}, \bar{q}_b^{\ddagger}, ..., \bar{q}_y^{\ddagger}\}$ and $\bar{e}^* \in \bar{\Sigma}$, $\Psi_{\bar{\Sigma}}(\bar{e}^*) = \{\bar{e}^*\}$, then
$\bar{K}(\Lambda_{\bar{Q}}(\bar{q}^{\ddagger}), \bar{e}^*) = \tilde{K}(\Lambda_{\bar{Q}}(\bar{q}_a^{\ddagger}), \bar{e}^*)$. $\tilde{K}(\Lambda_{\bar{Q}}(\bar{q}_b^{\ddagger}), \bar{e}^*)$. .. .$\tilde{K}(\Lambda_{\bar{Q}}(\bar{q}_y^{\ddagger}), \bar{e}^*)$. If $\bar{e}^*$ is not connected to any element of $\Psi_{\bar{Q}}(\bar{q}^{\ddagger})$, then $\bar{K}(\Lambda_{\bar{Q}}(\bar{q}^{\ddagger}), \bar{e}^*) = 1\ .\ 1\ \ ...\ .1 = 1$. Let $\bar{q}_l^{\ddagger}$ and $\bar{e}^*$ in the lth subautomaton (the elements of $\Psi_{\bar{Q}}(\bar{q}^{\ddagger}) \setminus \{\bar{q}_l^{\ddagger}\}$ are in the other subautomata, $\tilde{K}(\Lambda_{\bar{Q}}(\dot{q}), \bar{e}^*) = 1$, for $\dot{q} \in \Psi_{\bar{Q}}(\bar{q}^{\ddagger}) \setminus \{\bar{q}_l^{\ddagger}\}$). In this case, $\bar{K}(\Lambda_{\bar{Q}}(\bar{q}^{\ddagger}), \bar{e}^*) = 1\ ....\ 1\ .\bar{K}_l(\Lambda_{\bar{Q}_l}(\bar{q}_l^{\ddagger}), \bar{e}^*)\ .1\ ...1 = \bar{K}_l(\Lambda_{\bar{Q}_l}(\bar{q}_l^{\ddagger}), \bar{e}^*)$ is obtained. Here, if $\hat{q}$, which is obtained by using $\bar{e}^*$ from $\bar{q}_l^{\ddagger}$, is an element of $\tilde{\mathcal{G}}$ then $\bar{K}_l(\Lambda_{\bar{Q}_l}(\bar{q}_l^{\ddagger}), \bar{e}^*) = 0$ $[\bar{K}(\Lambda_{\bar{Q}}(\bar{q}^{\ddagger}), \bar{e}^*) = 0]$. Otherwise, $\bar{K}_l(\Lambda_{\bar{Q}_l}(\bar{q}_l^{\ddagger}), \bar{e}^*) = 1$ $[\bar{K}(\Lambda_{\bar{Q}}(\bar{q}^{\ddagger}), \bar{e}^*) = 1]$.

iv) If $\bar{q}^{\ddagger} \in \bar{Q}$, $\Psi_{\bar{Q}}(\bar{q}^{\ddagger}) = \{\bar{q}_a^{\ddagger}, \bar{q}_b^{\ddagger}, ..., \bar{q}_y^{\ddagger}\}$ and $\bar{e}^* \in \bar{\Sigma}$, $\Psi_{\bar{\Sigma}}(\bar{e}^*) = \{\bar{e}_a^*, \bar{e}_b^*, ..., \bar{e}_x^*\}$, then $\bar{K}(\Lambda_{\bar{Q}}(\bar{q}^{\ddagger}), \bar{e}^*)$ $= 1\ ....\ 1 = 1$ since any repeated state is not element of $\tilde{\mathcal{G}}$.

Note that, each element of $\bar{\mathcal{G}}$ is also an element of $\tilde{\mathcal{G}}$ because of overlapping decompositions and expansions approach and $\bar{q} \in \bar{\mathcal{G}} \Rightarrow \bar{q} \in Q_j$, for $j \in \{1,...,\alpha\}$. Thus, $\bar{q}$ is an element of $\bar{\mathcal{G}}_j$ and also $\bar{q} \in \tilde{\mathcal{G}}$. This decentralized controller prevents the forbidden states in $\bar{\mathcal{A}}_T^K$. □

We can obtain a decentralized controller for the original automaton as follows:

$$K(\Lambda_Q(q), e) = \bar{K}(\Lambda_{\bar{Q}}(q), e), \quad q \in Q,\ e \in \Sigma \tag{7}$$

This controller is added to the state equation (1) and $S(\tau) = (\mathcal{C} \vee S(\tau_e)) \wedge (\mathcal{O}(e, \tau_e) \otimes K(S(\tau_e), e))$, for $e \in \Sigma$ is obtained. It is known that, although the forbidden states are elements of Q, the new states may be elements of $\bar{\mathcal{G}}$. The controller of the OAA only disables the elements of Σ because of the connection of the pairs new states and events (see, Section 4). Therefore, the occurence of any event, which is disabled at any state by the controller (6), is disabled for the original automaton by the controller (7).

5 Example

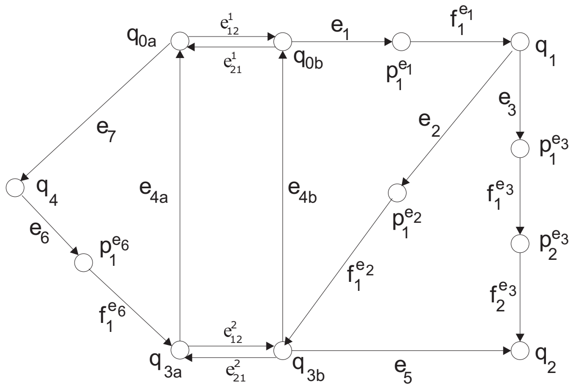

Figure 3. Expanded automaton

In this section, we design a decentralized controller, which guarantees the forbidden states avoidance, for the given timed automaton (Fig. 1a). The augmented automaton for this automaton is obtained as Fig. 2. The EAA, shown in Fig. 3, is obtained by using overlapping decompositions and expansions.

For the original timed automaton, the set of forbidden states is given as $\mathfrak{F} = \{q_2\}$ and $\bar{\mathfrak{F}} = \mathfrak{F}$. Then, $\tilde{\mathfrak{F}} = \bigcup_{\bar{q}\in\bar{\mathfrak{F}}} \Psi_{\bar{Q}}(\bar{q}) = \{q_2\}$. Now, we consider two subautomata to design a decentralized controller.

In the first subautomaton, $\bar{A}_{1_T}(\bar{Q}_1, \bar{\Sigma}_1, \bar{\mathfrak{C}}_1, q_{1_0})$, the set of states is $\bar{Q}_1 = \{q_{0a}, q_{3a}, q_4, p_1^{e_6}\}$, the set of events is $\bar{\Sigma}_1 = \{e_4, e_{6a}, e_7, f_1^{e_6}\}$, and the initial state is $q_{1_0} = q_{0a}$. In the second subautomaton, $\bar{A}_{2_T}(\bar{Q}_2, \bar{\Sigma}_2, \bar{\mathfrak{C}}_2, q_{2_0})$, the set of states is $\bar{Q}_2 = \{q_{0b}, q_2, q_{3b}, p_1^{e_1}, p_1^{e_2}, p_1^{e_3}, p_2^{e_3}\}$, the set of events is $\bar{\Sigma}_2 = \{e_1, e_2, e_3, e_{4b}, f_1^{e_1}, f_1^{e_2}, f_1^{e_3}, f_2^{e_3}\}$, and the initial state is $q_{2_0} = q_{0b}$. The connection matrices are

$$\bar{\mathfrak{C}}_1 = \begin{bmatrix} 0 & e_{4a} & 0 & 0 \\ 0 & 0 & 0 & f_1^{e_6} \\ e_7 & 0 & 0 & 0 \\ 0 & 0 & e_6 & 0 \end{bmatrix} \quad \text{and} \quad \bar{\mathfrak{C}}_2 = \begin{bmatrix} 0 & 0 & 0 & e_{4b} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & f_1^{e_1} & 0 & 0 & 0 \\ 0 & 0 & 0 & e_5 & 0 & 0 & 0 & f_2^{e_3} \\ 0 & 0 & 0 & 0 & 0 & f_1^{e_2} & 0 & 0 \\ e_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & e_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & e_3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & f_1^{e_3} & 0 \end{bmatrix}.$$

For each subautomaton, $\bar{\mathfrak{F}}_i = \tilde{\mathfrak{F}} \cap \bar{Q}_i$ for $i \in \{1, 2\}$ is obtained such as $\bar{\mathfrak{F}}_1 = \emptyset$ and $\bar{\mathfrak{F}}_2 = \{q_2\}$. In the subautomata, the set $\bar{\mathfrak{G}}_1 = \emptyset$ is obtained by using the algorithm *FS* (note that, $\bar{\mathfrak{G}}_{1_v} = \emptyset$). Thus, $\bar{K}_1(\Lambda_{\bar{Q}_1}(q_{0a}), \bar{e}^+) = \bar{K}_1([1\ 0\ 0\ 0]^T, \bar{e}^+) = 1$ for all $\bar{e}^+ \in \bar{\Sigma}_1$ and $\bar{K}_1(\Lambda_{\bar{Q}_1}(\bar{q}^*), \bar{e}^x) = 1$ for all $\bar{q}^* \in \bar{Q}_1$ and $\bar{e}^x \in \bar{\Sigma}_1$. The set $\bar{\mathfrak{G}}_2 = \{q_2, p_2^{e_3}, p_1^{e_3}\}$ is obtained by using the algorithm *FS* (note that,
$\bar{\mathfrak{G}}_{2_v} = \{[0\ 1\ 0\ 0\ 0\ 0\ 0\ 0]^T, [0\ 0\ 0\ 0\ 0\ 0\ 0\ 1]^T, [0\ 0\ 0\ 0\ 0\ 0\ 1\ 0]^T\}$). Thus, $\bar{K}_2(\Lambda_{\bar{Q}_2}(q_1), e_5) = 0$, $\bar{K}_2(\Lambda_{\bar{Q}_2}(q_{3a}), e_3) = 0$ and $\bar{K}_2(\Lambda_{\bar{Q}_2}(\bar{q}^x), \bar{e}^*) = 1$ for all $\bar{q}^x \in \bar{Q}_2 \setminus \{q_1, q_{3a}\}$ and $\bar{e}^* \in \bar{\Sigma}_2$.

Using the equation (5), the controller is obtained for the EAA as $\tilde{K}(\Lambda_{\tilde{Q}}(q_1), e_5) = \bar{K}_2(\Lambda_{\bar{Q}_2}(q_1), e_5) = 0$, $\tilde{K}(\Lambda_{\tilde{Q}}(q_{3b}), e_3) = \bar{K}_2(\Lambda_{\bar{Q}_2}(q_{3b}), e_3) = 0$, and $\tilde{K}(\Lambda_{\tilde{Q}}(\tilde{q}^\ddagger), \tilde{e}^c) = 1$, $\forall \tilde{q}^\ddagger \in \tilde{Q} \setminus \{q_1, q_{3a}\}$, $\forall \tilde{e}^c \in \tilde{\Sigma}$. It is known that $\tilde{S}_n = \Lambda_{\tilde{Q}}(q_1) = [0\ 1\ 0\ 0\ 0\ 0\ 0\ 0; 0\ 0]^T$, and $\Lambda_{\bar{Q}_2}(q_1) = [0\ 1\ 0\ 0\ 0\ 0\ 0\ 0]^T$.

Finally, a decentralized controller, which avoids the forbidden states, is designed by using (6). This controller is given as
$\bar{K}(\Lambda_{\bar{Q}}(q_1), e_5) = \tilde{K}(\Lambda_{\tilde{Q}}(q_1), e_5) = \bar{K}_2(\Lambda_{\bar{Q}_2}(q_1), e_5) = 0$, $\bar{K}(\Lambda_{\bar{Q}}(q_3), e_3) = \tilde{K}(\Lambda_{\tilde{Q}}(q_{3a}), e_3) \,.\, \tilde{K}(\Lambda_{\tilde{Q}}(q_{3b}), e_3) = \bar{K}_1(\Lambda_{\bar{Q}_1}(q_{3a}), e_3) \,.\, \bar{K}_2(\Lambda_{\bar{Q}_2}(q_{3b}), e_3) = 1 \,.\, 0 = 0$, and $\bar{K}(\Lambda_{\bar{Q}}(q^d), e^w) = 1$, $\forall q^d \in \bar{Q} \setminus \{q_1, q_3\}$, $\forall e^w \in \bar{\Sigma}$. The final result is given such that the occurence of e_5 is disabled at the state q_1 and the occurence of e_3 is

disabled at the state q_3. Thus, decentralized controller avoids state q_2. For the original timed automaton, the controller is obtained as $K(\Lambda_Q(q_3), e_3) = 0$, $K(\Lambda_Q(q_1), e_5) = 0$, and $K(\Lambda_Q(q^x), e^z) = 1$, $\forall q^x \in Q \setminus \{q_1, q_3\}$, $\forall e^z \in \Sigma$.

Now, let us compare the results of centralized and decentralized controllers for the given automaton. Both of these controllers disables the occurence of e_5 the state q_1 and the occurence of e_3 at the state q_3. The most advantage is that the size of the connection matrix for each subautomaton is smaller then the size of the connection matrix of the OAA.

6 Conclusion

A decentralized controller approach using overlapping decompositions for the timed discrete event systems. An approach, called *augmentation*, is presented to obtain the new modelling method such that each unit delay of any event represents a pair of new state and event. The augmented automaton is constructed by adding the pairs of events and states to the original automaton in this work.

The decentralized controller design approach is presented to prevent the occurence of the forbidden states. The augmented automaton is first decomposed overlappingly and expanded to obtain subautomata. Then, a controller is designed for each disjoint subautomaton. These local controllers are then combined to obtain a controller for the augmented automaton. Moreover, the state space representation is used for timed and untimed automata by the given algebraic approach.

Since the clock or timer does not used to analyse for the augmented automaton, the first advantage is that the computational complexity does not depend on clock for the timed automata. For the construction of the augmented autonmaton, the new states and events are added to the original automaton, and then the size of the connection matrix of the original automaton is smaller than the size of the connection matrix of the augmented automaton. Although this seems to be a disadvantage, the connection matrices of the augmented subautomata are only used to design the decentralized controller (i.e., the connection matrix of the augmented automaton is not used for the decentralized controller design approach). The size of the connection matrix of each subautomaton is an advantage for the decentralized approach (the number of states and events of subautomata is less than original automaton, [14, 15]).

Although the effort needed to obtain a useful the overlapping decomposition, this can be not comparable to the controller design since the decomposition may, in most cases, be easily made. Further research can also be undertaken to use this approach to design decentralized controllers for various objectives (for example, a controller can be designed such that this controller leads the given discrete event systems to marked states).

Bibliography

[1] P. J. G. Ramadge and W. M. Wonham, "The control of discrete event systems," *Proceedings of the IEEE*, vol. 77, pp. 81–98, 1989.

[2] R. S. Sreenivas and B. H. Krogh, "On Petri net models of infinite state supervisors," *IEEE Transactions on Automatic Control*, vol. 37, pp. 274–277, 1992.

[3] A. Aybar and A. İftar, "Decentralized supervisory controller design to avoid deadlock in Petri nets," *International Journal of Control*, vol. 76, pp. 1285–1295, 2003.

[4] 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)*, vol. 11, pp. 553–568, 2004.

[5] A. A. Desrochers and R. Y. Al-Jaar, *Applications of Petri Nets in Manufacturing Systems*, The Institute of Electrical and Electronics Engineers Inc., New York, 1995.

[6] M. Zhou and F. DiCesare, *Petri Net Synthesis for Discrete Event Control of Manufacturing Systems*, Kluwer Academic, Norwell, MA, 1993.

[7] R. Alur and D. L. Dill, “A theory of timed automata,” *Theoretical Computer Science*, vol. 126, pp. 183–235, 1994.

[8] A. Gouin and J. Ferrier, “Temporal coherence of timed automata product,” in *Proc. of the 1999 IEEE International Conference on Systems, Man, and Cybernetics*, October 1999, pp. 176–181.

[9] J. Krakora, L. Waszniowski, P. Pisa, and Z. Hanzalek, “Timed automata approach to real time distributed system verification,” in *Proc. of the 2004 IEEE International Workshop on Factory Communication Systems*, September 2004, pp. 407–410.

[10] A. Khoumsi, “A supervisory control method for ensuring the comformance of real-time discrete event systems,” *Discrete Event Dynamic Systems: Theory and Applications*, vol. 15, pp. 397–431, 2005.

[11] B. A. Bradin and W. M. Wonham, “Supervisory control of timed discrete–event systems,” *IEEE Transactions on Automatic Control*, vol. 39, pp. 329–342, 1994.

[12] F. Lin and W. M. Wonham, “Supervisory control of timed discrete–event systems under partial observation,” *IEEE Transactions on Automatic Control*, vol. 40, pp. 558–562, 1995.

[13] I. Açıksöz, “Time step approach for timed automata model (in turkish),” M.S. thesis, Anadolu University, Eskişehir, Turkey, June 2006.

[14] A. Aybar and A. İftar, “Overlapping decompositions of large–scale discrete–event systems,” in *Proceeding CD-ROM of The 15th IFAC World Congress*, Barcelona, Spain, July 2002.

[15] K. Rudie and W. M. Wonham, “Think globally, act locally: decentralized supervisory control,” *IEEE Transactions on Automatic Control*, vol. 37, pp. 1692–1708, 1992.

[16] A. Aybar and A. İftar, “Supervisory controller design for timed Petri nets,” in *Proceedings of the IEEE International Conference on System of Systems Engineering*, Los Angeles, CA, U.S.A., Apr. 2006, pp. 59–64.

[17] A. Aybar and A. İftar, “Deadlock avoidance controller design for timed Petri nets using stretching,” *IEEE Systems Journal*, vol. 2, pp. 178–188, 2008.

[18] M. Ikeda and D. D. Šiljak, “Overlapping decompositions, expansions, and contractions of dynamic systems,” *Large Scale Systems*, vol. 1, pp. 29–38, 1980.

[19] A. Aybar and A. İftar, “Overlapping decompositions and expansions of Petri nets,” *IEEE Transactions on Automatic Control*, vol. 47, pp. 511–515, 2002.

[20] A. Aybar, A. İftar, and H. Apaydın-Özkan, “Centralized and decentralized supervisory controller design to enforce boundedness, liveness, and reversibility in Petri nets,” *International Journal of Control*, vol. 78, pp. 537–553, 2005.

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