Localization of DES Supervisory Control with Event Reduction

Vahid Saeidi\(^1\), Ali A. Afzalian\(^2\) and Davood Gharavian\(^3\)
Department of Electrical Eng., Abbspour School of Engineering,
Shahid Beheshti University, Tehran, Iran
\(^1\)v_saeidi@sbu.ac.ir, \(^2\)Afzalian@sbu.ac.ir, \(^3\)d_gharavian@sbu.ac.ir

Abstract

Supervisor localization procedure can be employed to construct local controllers corresponding to component agents in discrete-event systems. This proposed method in \cite{11} is based on state reduction of a monolithic supervisor with respect to each set of controllable events corresponding to each component agent. A supervisor is localizable if state cardinality can be reduced from the reduced supervisor to each local controller. Although event reduction is an important property, the original supervisor localization procedure did not guarantee event reduction in each local controller comparing to the reduced supervisor. In this paper, we propose a method to localize a supervisor with event reduction in each local controller comparing to the reduced supervisor. State reduction facilitates the implementation of local controllers on industrial systems, whereas event reduction reduces communication traffic between each pair of local controllers.

Key words: control equivalent, event reduction, state reduction, supervisor localization, supervisor reduction.

1. Introduction

The supervisory control theory (SCT) encounters with some issues such as discrete-event modeling and computational complexity \cite{1}. Modular \cite{2-5} and incremental/compositional \cite{6-9} approaches try to overcome the complexity of the supervisor synthesis. Other approaches tend to reduce the supervisor for simple implementation \cite{10}. Such approaches do not affect on computational complexity reduction.

Supervisor localization procedure, introduced in \cite{11}, is a method to distribute the supervisory control of discrete-event systems. This procedure achieves two goals: i) it preserves the optimality (i.e. minimally restrictive) and non-blocking of the monolithic supervisor, and ii) it dramatically simplifies each local controller based on the state size criterion. Namely, a supervisor is localizable if the state size of each local controller is less than the state size of the reduced supervisor. Both goals are achieved by a suitable extension of supervisor reduction procedure \cite{10}. This procedure is carried out using control information relevant to the target agent. As the result, each agent obtains its own local controller. However, the control authority of a local controller is strictly local; the observation scope of each local controller is systematically determined in order to guarantee the correct local control action. This procedure cannot guarantee event reduction in each local controller comparing to the
reduced supervisor. This procedure has been extended in [12], for localization of the supervisory control under partial observation and using relative observability property [13]. Some applications of supervisor localization can be found in [14].

On the other hand, decomposition of a supervisor [15] is an alternative method to reduce the number of events, considered in decision making by each decentralized supervisor. This method constructs a distributed supervisory control with restricted control authority and restricted observation scope.

In this paper, we propose a supervisor localization procedure which ensures the event reduction from the reduced supervisor to each local controller.

The main common concepts in the supervisor reduction/localization procedures are control consistency of states. A cover on the set of states is constructed so that at least a pair of states which are not control consistent with respect to all controllable events but are control consistent with respect to a subset of controllable events corresponding to a component of the plant belongs to a subset of states which are lumped. Also, the next state by such transitions from a pair of states in the plant correspondingly should not be the same.

It is proved that, each local controller which is constructed by the proposed method has less number of events comparing to the reduced supervisor (in the sense of normality [16, 17]). It is an important result. However, event reduction in each local controller does not guarantee decomposability of a supervisor; each local controller needs fewer events for consistent decision making, comparing to the reduced state supervisor. It shows reducing the communication traffic between local controllers in a distributed supervisory control.

An induced generator is constructed by substituting one state for each subset in the control cover. The constructed induced generator must be a deterministic automaton. The set of constructed induced generators is control equivalent to the supervisor with respect to (w.r.t.) the plant.

In Section 2, the necessary preliminaries are reviewed. In Section 3, we give a short review on supervisor reduction procedure. In Section 4 we propose a method to localize the supervisory control with event reduction. Finally, conclusions and future work are given in Section 5.

2. Preliminaries

A discrete-event system is presented by an automaton $G = (Q, \Sigma, \delta, q_0, Q_m)$, where $Q$ is a finite set of states, with $q_0 \in Q$ as the initial state and $Q_m \subseteq Q$ being the marked states; $\Sigma$ is a finite set of events ($\sigma$) which is partitioned as a set of controllable events $\Sigma_c$, and a set of uncontrollable events $\Sigma_u$, where $\Sigma = \Sigma_c \cup \Sigma_u$. $\delta$ is a transition mapping $\delta: Q \times \Sigma \rightarrow Q$, $\delta(q, \sigma) = q'$ gives the next state $q'$ is reached from $q$ by the occurrence of $\sigma$. In this context $\delta(q_0, s)$ means that $\delta$ is defined for $s$ at $q_0$. $L(G) := \{s \in \Sigma^* | \delta(q_0, s)\}^*$ is the closed behavior of $G$ and $L_m(G) := \{s \in L(G) | \delta(q_0, s) \in Q_m\}$ is the marked behavior of $G$ [18, 19]. In this paper, we assume that $G$ consists of component agents $G^k$, defined on pairwise disjoint events sets $\Sigma^k$, $\forall k \in \mathcal{K}$, $\mathcal{K}$ is an index set, i.e. $\Sigma = \bigsqcup \{\Sigma^k | k \in \mathcal{K}\}$. Let $L_k := L(G^k)$ and $L_m,k := L_m(G^k)$, the closed and marked behaviors of $G$ are $L(G) = \|
\{L_k | k \in \mathcal{K}\} \text{ and } L_m(\mathbf{G}) = \| \{L_{m,k} | k \in \mathcal{K}\}, \text{ respectively, where } \| \text{ denotes synchronous product [19]. We assume that for every } k \in \mathcal{K}, \bar{L}_{m,k} = L_k \text{ is true. Then } \mathbf{G} \text{ is necessarily non-blocking (i.e. } \bar{L}_m(\mathbf{G}) = L(\mathbf{G})). \text{ A set of all control patterns is denoted with } \Gamma = \{\gamma \in Pwr(\Sigma) | \gamma \supseteq \Sigma_u\}. \text{ A supervisor of a plant } \mathbf{G}, \text{ is a map } V: L(\mathbf{G}) \to \Gamma, \text{ where } V(s) \text{ represents the set of enabled events after the occurrence of the string } s \in L(\mathbf{G}). \text{ A pair } (\mathbf{G}, V) \text{ is written by } V / \mathbf{G} \text{ and called } \mathbf{G} \text{ is under supervision by } V. \text{ A behavioral constraint on } \mathbf{G} \text{ is given by specification language } E \subseteq \Sigma^*. \text{ Let } K \subseteq L_m(\mathbf{G}) \cap E \text{ be the supremal controllable sublanguage of } E \text{ w.r.t. } L(\mathbf{G}) \text{ and } \Sigma_u, \text{ i.e. } K = \sup \mathcal{C} (L_m(\mathbf{G}) \cap E). \text{ If } K \neq \emptyset, \text{ SUP } = (X, \Sigma, \xi, x_0, X_m) \text{ is recognizer of } K. \text{ Write } |.| \text{ for the state size of DES. Then } |\text{SUP}| \leq |G||E|. \text{ In applications, engineers want to employ the reduced supervisor } \text{RSUP}, \text{ which has a fewer number of states and is control equivalent to SUP w.r.t. } G [10], \text{ i.e.}

\begin{align}
L_m(\mathbf{G}) \cap L_m(\text{RSUP}) &= L_m(\text{SUP}), \\
L(\mathbf{G}) \cap L(\text{RSUP}) &= L(\text{SUP}).
\end{align}

A generator \text{LOC}^k \text{ over } \Sigma, \text{ is a local controller for agent } \mathbf{G}^k, \text{ if } \text{LOC}^k \text{ can disable only events in } \Sigma_c^k, \text{ where } \Sigma_c^k = \Sigma^k \cap \Sigma_c. \text{ Precisely, for all } s \in \Sigma^* \text{ and } \sigma \in \Sigma, \text{ there holds,}

s\sigma \in L(\mathbf{G}) \& \ s \in L(\text{LOC}^k) \& \ s\sigma \notin L(\text{LOC}^k) \Rightarrow \sigma \in \Sigma_c^k.

The observation scope of \text{LOC}^k \text{ is not limited to } \Sigma^k. \text{ But, the control authority of a local controller is strictly local [11]. A set of local controllers } \text{LOC} = \{\text{LOC}^k | k \in \mathcal{K}\} \text{ is constructed, each one for an agent, with } L(\text{LOC}) = \cap \{L(\text{LOC}^k) | k \in \mathcal{K}\} \text{ and } L_m(\text{LOC}) = \cap \{L_m(\text{LOC}^k) | k \in \mathcal{K}\} \text{ such that the following relationships hold,}

\begin{align}
L_m(\mathbf{G}) \cap L_m(\text{LOC}) &= L_m(\text{SUP}), \\
L(\mathbf{G}) \cap L(\text{LOC}) &= L(\text{SUP}).
\end{align}

We say that, \text{LOC} \text{ is control equivalent to SUP w.r.t. } G, \text{ if (3) and (4) are satisfied. This formulation is based on state reduction of a monolithic supervisor with respect to disabled controllable events of each component agent.}

The natural projection is a mapping \text{P}: \Sigma^* \to \Sigma_o^*, \text{ where (1) } \text{P}(e) = e, (2) \text{ for } s \in \Sigma^*, \sigma \in \Sigma, \text{P}(s\sigma) = P(s)P(\sigma), \text{ and (3) } \text{P}(\sigma) = \sigma \text{ if } \sigma \in \Sigma_0 \text{ and } P(\sigma) = e \text{ if } \sigma \notin \Sigma_0. \text{ The effect of an arbitrary natural projection } \text{P} \text{ on a string } s \in \Sigma^* \text{ is to erase the events in } s, \text{ that do not belong to observable events set, } \Sigma_0. \text{ The natural projection } \text{P} \text{ can be extended and denoted by } \text{P}: Pwr(\Sigma^*) \to Pwr(\Sigma_o^*). \text{ For any } L \subseteq \Sigma^*, \text{P}(L) = \{P(s) | s \in L\}. \text{ The inverse image function of } \text{P} \text{ is denoted by } P^{-1}: Pwr(\Sigma_o^*) \to Pwr(\Sigma^*) \text{ for any } L \subseteq \Sigma_o^*, P^{-1}(L) = \{s \in \Sigma^* | P(s) \subseteq L\}. \text{ K is relative observable w.r.t. } \tilde{\mathcal{C}}, \mathbf{G} \text{ and } \text{P}, \text{ for } K \subseteq \mathcal{C} \subseteq L_m(\mathbf{G}), \text{ where } \tilde{\mathcal{C}} \text{ and } \tilde{\mathcal{C}} \text{ are prefix closed languages, if for every pair of strings } s, s' \in \Sigma^* \text{ such that } P(s) = P(s'), \text{ the following two conditions hold [13],}

\begin{align}
(a) \ (\forall \sigma \in \Sigma) \ s\sigma \in \bar{K}, s' \in \tilde{\mathcal{C}} \Rightarrow s\sigma \in L(\mathbf{G}) \Rightarrow s'\sigma \in \bar{K}, \\
(b) \ s \in K, s' \in \tilde{\mathcal{C}} \cap L_m(\mathbf{G}) \Rightarrow s' \in K.
\end{align}

In the special case, if \mathcal{C} = K, \text{ then the relative observability property is tighten to the observability property. An observation property called normality was defined in [16], that is stronger than the relative observability. K is said to be normal w.r.t. } (L(\mathbf{G}), \text{P}),
if \( P^{-1}P(\overline{R}) \cap L(G) = \overline{R} \), where \( L(G) \) is a prefix closed language and \( P \) is a natural projection.

3. Supervisor reduction procedure

A procedure was proposed in [10], to reduce the state size of a monolithic supervisor. This method is employed to construct a generator, which is control equivalent to the monolithic supervisor w.r.t. the plant. Let \( \text{SUP} = (X, \Sigma, \xi, x_0, X_m) \) and define \( E: X \to P\text{wr}(\Sigma) \) as \( E(x) = \{ \sigma \in \Sigma | \xi(x, \sigma) \} \). \( E(x) \) denotes the set of events enabled at state \( x \). Next, define \( D: X \to P\text{wr}(\Sigma) \) as \( D(x) = \{ \sigma \in \Sigma | -\xi(x, \sigma) \} \& (\exists s \in \Sigma^*)[\xi(x_0, s) = x \& \delta(q_0, s) \} \}. \( D(x) \) is the set of events, which are disabled at state \( x \). Define \( M: X \to \{0,1\} \) according to \( M(x) = 1 \) iff \( x \in X_m \), namely, flag \( M \) determines if a state is marked in \( \text{SUP} \). Also, define \( T: X \to \{0,1\} \) according to \( T(x) = 1 \) iff \( (\exists s \in \Sigma^*)[\xi(x_0, s) = x \& \delta(q_0, s) \} \} \), namely, flag \( T \) determines if some corresponding state is marked in \( G \). Let \( R \subseteq X \times X \) be the binary relation such that for \( x, x' \in X \), \( (x, x') \in R \). \( x \) and \( x' \) are called control consistent, if

\[
E(x) \cap D(x') = E(x') \cap D(x) = \emptyset, \quad (5)
\]

\[
T(x) = T(x') \Rightarrow M(x) = M(x'). \quad (6)
\]

While \( R \) is reflexive and symmetric, it need not be transitive, consequently it is not an equivalence relation. This fact underlies the next definition. A cover \( C = \{X_i \subseteq X | i \in I\} \) of \( X \) is called a control cover on \( \text{SUP} \) if \([10],
\]

\[
(\forall i \in I)X_i \neq \emptyset \land (\forall x, x' \in X_i)(x, x') \in R,
\]

\[
(\forall i \in I)(\forall \xi \in \Sigma)(\exists j \in I)[(\forall x \in X_i)\xi(x, \sigma) \Rightarrow \xi(x, \sigma) \in X_j], \quad (7)
\]

Where \( I \) is an index set. A control cover \( C \) lumps states of \( \text{SUP} \) into cells \( X_i \ (i \in I) \), if they are control consistent. A control cover \( C \) is control congruence if \( X_i \) are pairwise disjoint. Given control cover \( C = \{X_i \subseteq X | i \in I\} \) on \( \text{SUP} \), an induced supervisor is constructed as \( J = (I, \Sigma, \kappa, i_0, I_m) \), where \( i_0 \in I \) such that \( x_0 \in X_{i_0} \), \( I_m = \{i \in I | X_i \cap X_m = \emptyset \} \) and \( \kappa: I \times \Sigma \to I \) with \( \kappa(i, \sigma) = j \) provided, for such choice of \( j \in I, (\exists x \in X_j)[\xi(x, \sigma) \in X_j \& (\forall x' \in X_j)[\xi(x', \sigma) \Rightarrow \xi(x', \sigma) \in X_j]. \quad (9)\)

Overlapping of some states results that \( i_0 \) and \( \kappa \) may not be uniquely determined. If \( C \) is control congruence, then \( J \) is uniquely determined. Generally, \( J \) is control equivalent to \( \text{SUP} \) w.r.t \( G \). \( \text{RSUP} = (Z, \Sigma, \zeta, z_0, Z_m) \) is normal w.r.t \( \text{SUP} \) if,

\[
(i)(\forall z \in Z)(\exists s \in L(G))\zeta(z_0, s) = z,
\]

\[
(ii)(\forall z \in Z)(\forall \sigma \in \Sigma)[\zeta(z, \sigma)! \Rightarrow (\exists s \in L(G))] [s \sigma \in L(G) \& \zeta(z_0, s) = z], \quad (10)
\]

\[
(iii)(\forall z \in Z_m)(\exists s \in L_m(G))\zeta(z_0, s) = z.
\]

Given two generators \( \text{RSUP} \) and \( J \) are DES-isomorphic with isomorphism \( \theta \), if there exists a map \( \theta: Z \to I \) such that,
(i) $\theta: Z \to I$ is a bijection,
(ii) $\theta(z_0) = i_0$ and $\theta(Z_m) = l_m$,
(iii) $(\forall z \in Z)(\forall \sigma \in \Sigma) \xi(z, \sigma)! \Rightarrow 
[\kappa(\theta(z), \sigma)! \& \kappa(\theta(z), \sigma) = \theta(\xi(z, \sigma))],
(iv) (\exists i \in I)(\forall \sigma \in \Sigma) \kappa(i, \sigma)! \Rightarrow 
[(\exists z \in Z)(\xi(z, \sigma)!) \& \theta(\xi(z, \sigma)) = i].$

If $\text{SUP}$ is supremal supervisor for $G$ and $\text{RSUP}$ is any normal supervisor w.r.t $\text{SUP}$ such that it is control equivalent to $\text{SUP}$ w.r.t $G$, then there exists a control cover $C$ on $\text{SUP}$ for which some induced supervisor $J$ is DES-isomorphic to $\text{RSUP}$ [10].

4. Supervisor localization procedure with event reduction

Let $G = (Q, \Sigma, \delta, q_0, Q_m)$ be the plant and $\text{SUP} = (X, \Sigma, \xi, x_0, X_m)$ be the monolithic supervisor. Define $E$ be same as the one defined in Section 3. Next, define $D^k: X \to Pwr(\Sigma^k_c)$ as $D^k(x) = \{\sigma \in \Sigma^k_c \mid \neg \xi(x, \sigma)! \& (\exists s \in \Sigma^*)[\xi(x_0, s) = x \& \delta(q_0, s\sigma)!]\}$. Flags $M$ and $T$ are defined same as the ones defined in Section 3. Let $R^k \subseteq X \times X$ be the binary relation such that for $x, x' \in X, (x, x') \in R^k$. $x$ and $x'$ are called control consistent w.r.t. $\Sigma^k_c$, if

$$E(x) \cap D^k(x') = E(x') \cap D^k(x) = \emptyset, \tag{12}$$

$$T(x) = T(x') \Rightarrow M(x) = M(x'). \tag{13}$$

Before constructing a cover on $X$ we describe some conditions to satisfy event reduction in each local controller.

Assume $(\exists x, x' \in X), (x, x') \in R, (x, x') \in R^k, (\exists s, s' \in \Sigma^*), x = \xi(x_0, s), x' = \xi(x_0, s'),$ if $(\exists \sigma \in \Sigma), q = \delta(q_0, s\sigma)! , q' = \delta(q_0, s'\sigma)!$ then $q \neq q'$. \tag{14}

A cover $C^k = \{X^k_i \subseteq X | i \in I^k\}$ is a control cover on $X$ w.r.t. $\Sigma^k_c$ if (14), (15) and (16) are satisfied.

$$(\forall i^k \in I^k) X^k_{i^k} \neq \emptyset \wedge (\forall x, x' \in X^k_{i^k})(x, x') \in R^k, \tag{15}$$

$$(\forall i^k \in I^k)(\forall \sigma \in \Sigma)(\exists j^k \in I^k)[(\forall x \in X^k_{i^k})\xi(x, \sigma)! \Rightarrow \xi(x, \sigma) \in X^k_{j^k}], \tag{16}$$

Where $I^k$ is some index set. Given $C^k$ on $X$, based only on the control information of $\Sigma^k_c$, an induced generator $I^k = (I^k, \Sigma, \kappa^k, i_0^k, l_m^k)$ is obtained by the following construction,

$$(i) i_0^k \in I^k \text{ such that } x_0 \in X^k_{i_0^k},$$

$$(ii) l_m^k = \{I^k | X^k_{i^k} \cap X_m \neq \emptyset\},$$

$$(iii) \kappa^k: I^k \times \Sigma \to I^k \text{ with } \kappa^k(i^k, \sigma) = j^k, \text{if }$$

$$\exists x \in X^k_{i^k}\xi(x, \sigma) \in X^k_{j^k} \& (\forall x' \in X^k_{i^k})$$

$$\left[\xi(x', \sigma)! \Rightarrow \xi(x', \sigma) \in X^k_{j^k}\right].$$

We can obtain a set of induced generators $\text{LOC} = \{\text{LOC}^k | k \in K\}$. Let $L(\text{LOC}) := \cap \{L(\text{LOC}^k) | k \in K\}$ and $L_m(\text{LOC}) := \cap \{L_m(\text{LOC}^k) | k \in K\}$. $\text{LOC}$ is a solution to the distributed supervisory control problem with event reduction.

A DES $\text{LOC} = (Z_L, \Sigma, \xi_L, Z_{L,0}, Z_{L,m})$ is normal w.r.t $\text{SUP}$ if, (18) is satisfied.
In the following proposition, it is proved that each local controller, constructed by the proposed method in this paper, has at least one self-looped event at one state.

**Proposition 1:** Let $G$ be the non-blocking plant, which consists of components $G^k, k \in \mathcal{K}$, $\text{SUP}$ be the supervisor of $G$, and $\text{LOC}^k$ be the local controller corresponding to $G^k$. If $\exists x, x' \in X$, and $\exists \sigma \in \Sigma_i, j \neq k$ such that (14) does hold, then $\sigma$ is self-looped at a state in $\text{LOC}^k$, which is constructed by a subset of states that $x, x'$ belong to it.

**Proof:** Assume $x, x' \in X$, such that $x = \xi(x_0, s)$, $x' = \xi(x_0, s')$. According to (14), there exists $\sigma \in E(x) \cap D(x')$ or $\sigma \in E(x') \cap D(x)$. Assume $\sigma \in E(x) \cap D(x')$. Since $\sigma \not\in E(x) \cap D^k(x)$, $\sigma$ is disabled at $x'$ in $\text{SUP}$, but cannot be disabled there in $\text{LOC}^k$. On the other hand, $\sigma$ is defined at the corresponding state in $G$ and according to (14), if we can find $q, q'$ so that $q = \delta(q_0, s\sigma)!$, $q' = \delta(q_0, s'\sigma)!$, then $q \neq q'$. Hence, $\sigma$ must becomes a self loop transition at the state which is substituted for lumped states $x, x'$ in $\text{LOC}^k$. Otherwise, $\text{LOC}^k$ becomes nondeterministic automaton. Similarly, the proposition can be proved in the case of $\sigma \in E(x') \cap D(x)$.

**Corollary 1:** In Proposition 1, since $\sigma$ is disabled at some states of the supervisor, its observation affects the behavior of the supervisor. It means that, if $\sigma$ is not observed, then the supervisor makes inconsistent decisions.

In order to prove the main claim, we show that $\sigma$ appears as a self loop transition at all states of $\text{LOC}^k$. Since $\text{LOC}^k$ is not authorized to disable $\sigma$, it is sufficient to prove that $\sigma$ appears as a self loop transition at states, where it is enabled. In Lemma 1, we prove the claim in a special case.

**Lemma 1:** Let $G$ be the non-blocking plant, which consists of components $G^k, k \in \mathcal{K}$, and $\text{LOC}^k$ be a local controller corresponding to each $G^k$. Let $\sigma_0 \in \Sigma_i, j \neq k$, and one string belongs to $L(\text{LOC}^k)$ be as shown in Fig. 1. If $[(\exists s \in \Sigma^*), \delta(q_0, s\sigma_0)!, \neg\delta(q_0, s\sigma_1)!]; \exists i, z_i = \zeta_L(z_{L,0}, s) \Rightarrow z_i = \zeta_L(z_i, \sigma_0)]$, then $[\forall i, \zeta_L(z_i, \sigma_0)! \Rightarrow z_i = \zeta_L(z_i, \sigma_0)]$.

**Proof:** Assume the set of states and strings, are shown in Fig. 1. Define $P: \Sigma^* \rightarrow \Sigma_0$, $\Sigma_0 = \Sigma - \{\sigma_0\}$. We should prove that each pair of states, where $\sigma_0$ occurs in between, can be considered one state. We know that $P(s\sigma_0\sigma_1s'\sigma_0) = P(s\sigma_1s')$. Since a language, constructed by local controllers in the plant, is same as the language of the monolithic supervisor (see (3) and (4)), we can extend the observability property of the monolithic supervisor to each local controller. Thus, from Fig. 1, we can write,

$$\forall \sigma \in \Sigma, s\sigma_0\sigma_1s'\sigma_0 \sigma \in L(\text{LOC}^k) \cap L(G), s\sigma_1s' \in L(\text{LOC}^k), s\sigma_1s' \sigma \in L(G) \Rightarrow s\sigma_1s' \sigma \in L(\text{LOC}^k).$$
The string which occurs in both the local controller and the plant is shown in the first term of the antecedent in (21). Since \( s_{\sigma_1} \not\in L(\mathbf{G}) \), then \( s_{\sigma_1} s' \not\in L(\mathbf{G}) \). Thus, (21) is true. Similarly,

\[
s_{\sigma_0} s' \sigma_0 \in L_m(\mathbf{LOC}^k) \cap L_m(\mathbf{G}), \quad s_{\sigma_1} s' \in L(\mathbf{LOC}^k) \cap L_m(\mathbf{G}) \quad \Rightarrow \quad s_{\sigma_1} s' \in L_m(\mathbf{LOC}^k).
\]  

(20)

Since \( s_{\sigma_1} s' \not\in L(\mathbf{G}) \), then \( s_{\sigma_1} s' \not\in L_m(\mathbf{G}) \). Thus, (20) is true. Therefore, \( z_n \) and \( z_{n+1} \) can be considered one state, where \( \sigma_0 \) is a self-looped transition. However, the proposed supervisor localization procedure considers the plant cyclic (the task of the plant is assumed to be cyclic), even if it is not cyclic, we assume \( \mathbf{G} \) is a cyclic plant.

Thus, for \( \forall h, w \) "(\( x^g, s \))! \ we use the above argument and conclude that \( x^g = w \) "(\( x^g, s \)).

Since \( \mathbf{LOC}^k \) is not authorized to disable \( \sigma_0 \), it can be considered a self loop transition at other states, where \( \sigma_0 \) is not defined in \( \mathbf{G} \), correspondingly, i.e.

\[
(\exists s \in \Sigma^*) [\delta(q_0, s) = q \& z_i = \xi_L(z_{L,0}, s) \& \neg \delta(q, \sigma_0)!] \Rightarrow z_i = \xi_L(z_i, \sigma_0).
\]

Therefore, \( \sigma_0 \) is self-looped at all states in \( \mathbf{LOC}^k \).

Now, we relax the assumption \( s_{\sigma_1} \not\in L(\mathbf{G}) \), and show that Lemma 1 still does hold.

**Theorem 1:** Let \( \mathbf{G} \) be the non-blocking plant, which consists of components \( \mathbf{G}^k, k \in \mathcal{K} \), described by closed and marked languages \( L(\mathbf{G}), L_m(\mathbf{G}) \subseteq \Sigma^* \) and \( \mathbf{SUP} = (X, \Sigma, \xi, x_0, X_m) \) be the supervisor of \( \mathbf{G} \). If \( \exists x, x' \in X \), and \( \exists \sigma_0 \in \Sigma_c, j \neq k \) such that (14) does hold, then \( \sigma_0 \) is self-looped at all states in \( \mathbf{LOC}^k \).

**Proof:** Following the proof in Lemma 1, if \( s_{\sigma_1} \not\in L(\mathbf{G}) \), then (21) and (22) are true, in Fig. 1.

\[
s_{\sigma_0} s' \sigma_0 \sigma \in L(\mathbf{LOC}^k) \cap L(\mathbf{G}), \quad s_{\sigma_1} s' \in L(\mathbf{LOC}^k), \quad s_{\sigma_1} s' \sigma \in L(\mathbf{LOC}^k) \quad \Rightarrow \quad s_{\sigma_1} s' \sigma \in L(\mathbf{LOC}^k),
\]

(21)

\[
s_{\sigma_0} s' \sigma_0 \in L_m(\mathbf{LOC}^k) \cap L_m(\mathbf{G}), \quad s_{\sigma_1} s' \in L_m(\mathbf{LOC}^k) \cap L_m(\mathbf{G}) \quad \Rightarrow \quad s_{\sigma_1} s' \in L_m(\mathbf{LOC}^k).
\]

(22)

Now, we prove that (21) and (22) are true, even if \( s_{\sigma_1} \not\in L(\mathbf{G}) \). From Fig. 1, we know that \( s_{\sigma_1} \not\in L(\mathbf{LOC}^k) \). There may be two cases related to \( \sigma_1 \): (a) \( \sigma_1 \) is a transition from \( x \), or (b) \( \sigma_1 \) is a transition from \( x' \), in \( \mathbf{SUP} \).

(a) If \( \sigma_1 \) is enabled at \( x \), then \( \sigma_1 \) is enabled at \( x' \), or \( s' \sigma_1 \not\in L(\mathbf{G}) \). If \( s' \sigma_1 \not\in L(\mathbf{G}) \), then according to Lemma 1, \( \sigma_0 \) is self looped at all states of \( \mathbf{LOC}^k \).

Now, assume that \( \sigma_1 \) is enabled at \( x' \). Since \( \mathbf{LOC}^k \) is deterministic automaton, the next state from \( x \) and \( x' \) by transition \( \sigma_1 \) must be the same. It means that transition \( \sigma_1 \) is defined in \( \mathbf{G} \) from two different states \( q, q' \) to one state \( q_i \). From (14), it cannot be satisfied. Thus, the next states by transition \( \sigma_1 \) must be merged with \( x, x' \) and \( \sigma_1 \) is self-looped there. This argument can be continued for other subsequent states, until either all subsequent transitions become self-looped at the lumped states, in which

Fig. 1. A set of states and strings in \( \mathbf{LOC}^k \), where \( \sigma_0 \) is self-looped at \( z_i \)
$x, x'$ belongs to their subset (in this case, $\text{LOC}^k$ is reduced to one state with self loop transitions, and the claim is proved), or some enabled events can be found such that they are not defined at corresponding state in $G$. The latter was proved in Lemma 1.

It means that, there is a natural projection $P: \Sigma^* \to \Sigma_0^*, \Sigma_0 = \Sigma - \{\sigma_0\}$ and $P(s_0 \sigma_1 ... \sigma_{n-1}) = P(s \sigma_1 ... \sigma_{n-1})$, such that $\exists \sigma_n \in \Sigma, s \sigma_1 ... \sigma_{n-1} \sigma_n \notin L(G)$ and $s_0 \sigma_1 ... \sigma_{n-1} \sigma_n \in L(\text{LOC}^k) \cap L(G), s \sigma_1 ... \sigma_{n-1} \in L(\text{LOC}^k), s \sigma_1 ... \sigma_{n-1} \sigma_n \in L(G) \Rightarrow s \sigma_1 ... \sigma_{n-1} \sigma_n \in L(\text{LOC}^k)$.

(b) It can be proved similar to case (a), that $\sigma_0$ is self looped at all states in $\text{LOC}^k$.

We summarize the above argument in the following statement. If an event $\sigma_0$, which is not disabled in $\text{LOC}^k$, is self-looped at one state, then some other events may be self-looped there, until we can find some enabled events, such that they are not defined at corresponding state in $G$. Thus, there is a natural projection $P: \Sigma^* \to \Sigma_0^*, \Sigma_0 = \Sigma - \{\sigma_0\}$, such that $P(s) = P(s')$, $s \sigma \in L(\text{LOC}^k) \cap L(G), s' \sigma \in L(\text{LOC}^k), s \sigma \in L(G) \Rightarrow s' \sigma \in L(\text{LOC}^k), \text{ and } s' \sigma \notin L(G)$.

Similar to the method, proposed in [11], we prove that the local controllers, constructed by (15) - (18) are control equivalent to the monolithic supervisor w.r.t. the plant.

**Theorem 2:** Let $G$ be the non-blocking plant, which consists of components $G^k, k \in \mathcal{K}$, and $\text{SUP} = (X, \Sigma, \xi, x_0, X_m)$ be the supervisor of $G$. If $\text{LOC}$ is constructed by the procedure, proposed in (15) - (18), then $\text{LOC}$ and $\text{SUP}$ are control equivalent w.r.t. $G$.

**Proof:** This theorem can be proved similar to Proposition 1 in [11].

However, we prefer a set of local controllers, with fewer states than the reduced supervisor; the localization procedure in this paper guarantees less number of events in each local controller comparing to the reduced supervisor. In the next section we illustrate the proposed method by an example.

5. Examples- Localization of supervisory control of transfer line with event reduction

Industrial transfer line consists of two machines $M_1, M_2$ and a test unit TU, that are linked by buffers $B_1$ and $B_2$ with capacities 3 and 1, respectively (Fig. 2). If a work piece is accepted by TU, it is released from the system; if rejected, it is returned to $B_1$ for reprocessing by $M_2$. The specification is based on protecting $B_1$ and $B_2$ against underflow and overflow [19].

All events involved in the DES model are $\Sigma = \{1,2,3,4,5,6,8\}$, where controllable events are odd-numbered. After the synthesis of the supremal controllable supervisor, we obtain the reduced supervisor by $\text{supreduce}$ procedure in TCT software [20] (Fig. 3). Moreover, we construct local controllers for each component $M_1, M_2$ and TU, by the proposed procedure in this paper (Figs. 4-6).
Obviously, we see that each local controller has less number of states and less number of events, comparing to the reduced supervisor. In this example, the local controller $M_1$ does not use events 3, 4, 5, 8. Also, events 1, 6 are not used by the local controller $M_2$ and events 1, 2, 3, 6, 8 are not used by the local controller TU for decision making. Thus, the synthesized supervisor for industrial transfer line is localizable in terms of state reduction and event reduction criteria.

6. Conclusions

This paper addresses the new procedure for localization of the supervisory control, in which the local controllers have less number of events comparing to the reduced supervisor. In fact, this localization procedure provides fewer states, for easier implementation of each local controller on industrial systems, and provides fewer events, in order to reduce the communication traffic between local controllers. Reduction in the number of communicated events between local controllers is
important due to efficiency of a distributed supervisory control. We can only use the event reduction criterion for checking localizability of a supervisor. Localizability is satisfied by state reduction in each local controller, in the sense of [11], whereas, event reduction guarantees reduction in communication traffic between local controllers. By event reduction property, we can investigate and compare the localizability and decomposability of a supervisor, in future work.

References

[1] W. M. Wonham and P. J. Ramadge, "Modular supervisory control of discrete-event systems," Math. Control Signal Syst., vol.1, no.1, pp.13-30, 1988.
[2] G. Schafaschek, M. H. de Queiroz, J. E. R. Cury, "Local Modular Supervisory Control of Timed Discrete-Event Systems", WODES14, Cachan, May 2014, pp.271-277.
[3] L. Feng and W. M. Wonham, "Supervisory Control Architecture for Discrete-Event Systems", IEEE Trans. Autom. Control, vol.53, no.6, pp.1449-1461, 2008.
[4] Y. Willner and M. Heymann, "Supervisory control of concurrent discrete-event systems", International Journal of Control, vol.54, no.5, pp.1143-1169, 1991.
[5] F. Lin and W. M. Wonham, "Decentralized control and coordination of discrete-event systems with partial observation", IEEE Trans. Autom. Control, vol.35, no.12, pp.1330-1337, 1990.
[6] S. Mohajerani, R. Malik and M. Fabian, "Compositional synthesis of supervisors in the form of state machines and state maps", Automatica, vol.76, pp.277-281, 2017.
[7] R. Malik and M. Teixeira, "Modular Supervisor Synthesis for Extended Finite-State Machines Subject to Controllability", WODES16, Xi’an, June 2016, pp. 91-96.
[8] S. Mohajerani, R. Malik and M. Fabian, "A Framework for Compositional Synthesis of Modular Non-blocking Supervisors", IEEE Trans. Autom. Control, vol.59, no.1, pp.150-162, 2014.
[9] A. Vahidi, M. Fabian, B. Lennartson, "Efficient supervisory synthesis of large systems", Control Engineering Practice, vol.14, no.10, pp.1157-1167, 2006.
[10] R. Su and W. M. Wonham, "Supervisor reduction for discrete-event systems", Discrete-event Dyn. Syst., vol.14, no.1, pp.31-53, 2004.
[11] K. Cai and W. M. Wonham, "Supervisor Localization: A Top-Down Approach to Distributed Control of Discrete-Event Systems", IEEE Trans. Autom. Control, vol.55, no.3, pp.605-618, 2010.
[12] R. Zhang, K. Cai, W. M. Wonham, "Supervisor localization of discrete-event systems under partial observation", Automatica, vol.81, pp.142-147, 2017.
[13] K. Cai, R. Zhang, W.M. Wonham, "Relative Observability of Discrete-Event Systems and its Supremal Sublanguages", IEEE Trans. Autom. Control, vol.60, no.3, pp.659-670, 2015.
[14] K. Cai and W. M. Wonham, "New results on supervisor localization, with case studies", Discrete-event Dyn. Syst., vol.25, no.1, pp.203-226, 2015.
[15] K. Rudie and W. M. Wonham, "Think globally, act locally: decentralized supervisory control", IEEE Trans. Autom. Control, vol.37, no.11, pp.1692-1708, 1992.
[16] F. Lin, W. M. Wonham, "On observability of discrete-event systems", Information Sciences, vol.44, no.3, pp.173-198, 1988.
[17] R. Cieslak, C. Desclaux, A. S. Fawaz, P. Varaiya, "Supervisory control of discrete-event processes with partial observations", IEEE Trans. Autom. Control, vol.33, no.3, pp.249-260, 1988.
[18] C.G. Cassandras and S. Lafortune, "Introduction to Discrete-event Systems", 2nd edn. Springer, NewYork, 2008.
[19] W. M. Wonham, "Supervisory control of discrete-event systems", 2016, http://www.control.utoronto.ca/DES.
[20] W. M. Wonham, Control Design Software: TCT. Developed by Systems Control Group, University of Toronto, Canada, 2014, http://www.control.utoronto.ca/cgi-bin/dlxptct.cgi.