International Journal of Applied Engineering Research ISSN 0973-4562 Volume 10, Number 17 (2015) pp 37981-37991

Development of a coordination scheme in the framework of structural decentralized supervisory control of discrete-event system

Sang-Heon Lee,
Program director, The School of Engineering, The University of South Australia, Mawson Lakes, SA, Australia
sang-heon.lee@unisa.edu.au

Jae-Sam Park,
Professor, Department of Electronics Engineering Incheon National University Songdo Dong, Yeonsugu, Incheon, Korea
jaepark@inu.ac.kr

Abstract
A coordination scheme in the structural decentralized control of supervisory control theory (SCT) framework is developed in this paper. Generally, in many applications, after the decentralized control design is established, often minor changes of specifications are necessary to meet different requirements. In most cases, the conditions to establish such a coordination scheme with decentralized control need to be verified for the whole system again for every single specification change, resulting in large computational burden. Using the concept of a coordinator in a structural decentralized control framework, this paper shows that under the structural conditions, the combined control actions of the coordinator with the existing decentralized supervisors can achieve the same optimality as the centralized control and minor changes of specifications can be implemented without further verification. An example of chemical batch process is provided to illustrate the result..

Keywords: Task coordinator; Supervisory control system; Discrete-event system; Structural decentralized control; Simulation study

Introduction

Discrete event system (DES) describes the orderly changes of system behaviors without any information about the real time at which the changes occur. To analyze and design such systems, several formal methods are developed, and supervisory control theory (SCT) is one of such methods, proposed by Ramadge & Wonham [1] and further extended by other researchers [2, 3]. SCT is aimed to synthesize a supervisor to satisfy the specifications in an optimal or minimally restrictive way using the concept of the supremal controllable sublanguage [4]. Since the effort to compute a supervisor will grow exponentially with the increase of the number of components involved [5], the application of centralized approaches of DES into the real industrial practice is limited. In most cases, the purely centralized DES can be applicable to only small sized plants. In practice, a large DES can be subdivided into smaller local plants with communicating each other synchronously or asynchronously. Using this so called divide and conquer concept, within this framework, modular approach [6] and decentralized approach [7, 8, 9, 10] have been proposed and studied. However, in those approaches, the whole plant needs to be checked for the eligibility conditions to establish such schemes for every single specification given. Hence even though the computational complexity has been decreased a lot within those methods, still it was one of the main issues. Intuitively, however, if the system is properly structured with some flexibility, then the operation of such systems would be easier. A study conducted by Lee & Wong [11] accommodates this intuitive thinking as a core part of their structural decentralized scheme in the SCT framework. The fundamental idea underneath this approach is to arrange the structure of the whole plant to be most suitable for a certain set of decentralized operations. The conditions are dependent on the system structure not on each specification. Hence unlike other approaches, even though minor changes in the specification are introduced into the system, the conditions to establish such a scheme would not need to be verified again. That is the reason why this approach is called as a structural decentralized control. It has shown that this could bring an exponential savings on computational efforts in the long run while it still offers the same optimal behavior as that would be obtained by the centralized control.

This paper investigates an extension of this structural decentralized control into a coordination scheme. The concept of coordination in SCT is firstly introduced in Lin & Wonham [12] and Lin [13], in which the author(s) design decentralized supervisors with a coordinating supervisor (called coordinator) at a higher level to supervise the interactions with allocated tasks to achieve the overall tasks. The coordination scheme in SCT is further extended by the works of Komenda and others [14, 15]. Fundamentally, the coordination scheme is a compromised approach of hierarchical control with the higher level consisting of the coordinator and local supervisors. While the local supervisors are focused on the control of each corresponding local plant, the coordinator works more on the communication and interactions of behaviors among local plants. To establish successful coordination among the local plants, there are two main issues to be addressed: one is to ensure all corresponding local plants are to be ready before any synchronized coordination action happens, and the other is to ensure all local plants to be able to monitor any uncontrollable events (like breakdowns) occurring in other plants. Both conditions are necessary to guarantee nonblocking and nonconflicting control synthesis to achieve the overall specified tasks. Surprisingly these two conditions are similar to the conditions developed in Lee & Wong [11]. In addition again in all approaches in the

coordination scheme of SCT, the conditions to establish such schemes are specification dependent. The authors acknowledge that it is quite logical to synthesis the coordination scheme based on the given specifications. However, there is still a reasonable portion of systems that can be established such a coordination scheme in their structure for a set of specifications to provide higher flexibility, as shown in the example in Section 5 in this paper. This paper addresses this issue, namely, how concurrent actions of the coordinator and decentralized supervisors to be used to solve some coordination problems among structural decentralized plants with higher adaptability to various demands. We consider a situation in which after the structural decentralized control is established, a coordinating supervisor with synchronized events is synthesized to control the interactions of the local supervisors. Since the coordinator is established locally from the natural projection with synchronous compositions, the structure of the global plant will stay the same. For the specifications given in the coordination plant, which usually deals with the interactions or communications among local plants, a coordinator can be designed. As can be observed in the example provided in Section 5, this sort of arrangements might be common in chemical batch plants producing several different products using different combinations of materials and different sequences of operations with the same multi-purpose equipment [16]. The coordination scheme can be established so that each local supervisor controls each corresponding local plant to ensure the correct and orderly operation in the plant, while the coordinator would focus on the interactions among local plants, like necessary changes of combination of materials or changes of operation sequences. The conditions presented in this paper are a reasonable extension of those established in Lee & Wong [11]; the combined concurrent actions of the coordinator and the existing decentralized supervisors will solve a specific set of coordination requirements. Again we point out that since the conditions are applied in the system structure, hence once the structural coordination architecture has been verified and established, it can be used for a set of tasks without any further verification.

The remainder of the paper is organized as follows. Section 2 and 3 cover the basic notations and definitions of supervisory control theory and structural decentralized supervisory control framework, respectively. Section 4 presents the problem and the formulation of the main result (a coordination scheme) and Section 5 provides an example for the illustration of the results. Finally Section 6 presents the conclusions of the research.

Supervisory Control Theory

This section provides some definitions and formulations of supervisory control theory framework [2, 4, 17].

The behaviors of an uncontrolled DES process are modelled by a set of finite sequences of events. Let Σ denote a finite set of *event* labels, often called an *alphabet*. Then abstractly, a 5-tuple automaton, $\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_m)$, could represent the uncontrolled DES. In here, Q is a finite set of states, δ: $Q \times \Sigma \rightarrow Q$ is a (partial) *state transition function*, $q_0 \in Q$ is the *initial* state, and $Q_m \subseteq Q$ is a set of *marker* states. The events are the state transitions according to δ starting from the initial state q_0. Let Σ^* denote the set of all finite sets of event labels with the empty string, $\varepsilon \notin \Sigma$. Any subset of Σ^* is a *language* over Σ. Among those, all physically possible finite sequences of events that **G** can generate from the initial state q_0 is called the *closed behavior* of **G** (denoted $L(\mathbf{G})$) and a subset of $L(\mathbf{G})$ that reaches the marker states Q_m, representing completed tasks carried out by the physical process of **G**, is called the *marked behavior* of **G** (denoted $L_m(\mathbf{G})$). The formal definitions are respectively: $L(\mathbf{G}) = \{s \in \Sigma^* \mid \delta(q_0,s) \text{ is defined}\}$ and $L_m(\mathbf{G}) = \{s \in L(\mathbf{G}) \mid \delta(q_0,s) \in Q_m\}$.

The DES **G** simply allows any events defined in Σ^* to occur without any means of control. To implement control to such a DES, a supervisor needs to be designed so that the behavior of **G** is restricted to a desirable subset of states and transitions. By doing this, the supervisor ensures that the resulting closed-loop system behavior satisfies the specifications (like the right sequence of operations). To introduce a control to **G**, the entire events are divided into two: *controllable* events, $\Sigma_c \subseteq \Sigma$ and *uncontrollable* events, $\Sigma_u = \Sigma - \Sigma_c$. The controllable events are assumed to be those events that can be disabled (prevented from occurring) and enabled (permitted to occur) while the uncontrollable events are those always enabled (can happen at any time). By enabling or disabling controllable events according to the specifications E, the desired behavior of a given plant **G** can be generated by the supervisor **S**. Wonham & Ramadge (1987) have developed an algorithm to obtain such a supervisor:

$L_m(\mathbf{S/G}) = \kappa_{L(\mathbf{G})}(L_m(E) \cap L_m(\mathbf{G}))$.

Abstractly, $\kappa_{L(\mathbf{G})}$ represents the process to obtain the largest possible or optimal behavior that can be synthesized by a supervisor **S** for a plant **G** satisfying the given specification E. Within this framework, a *prefix* is defined as a string $s \in \Sigma^*$ for a string $v \in \Sigma^*$ such that $u = sv$, where $u \in \Sigma^*$. The *prefix closure* of $K \subseteq \Sigma^*$ is defined by $\overline{K} = \{s \in \Sigma^* \mid \exists v \in \Sigma^* \text{ such that } sv \in K\}$. Also $K \subseteq \Sigma^*$ is *closed* if $K = \overline{K}$. Let $F \subseteq K \subseteq \Sigma^*$. Then, the language F is said to be *K-closed* if $F = \overline{F} \cap K$. A language F is *nonblocking* with respect to K if $\overline{F} = \overline{F \cap K}$. Certainly if **G** satisfies $\overline{L_m(\mathbf{G})} = L(\mathbf{G})$, then $L(\mathbf{G})$ is non-blocking. Two languages H_1, $H_2 \subseteq \Sigma^*$ are said to be *nonconflicting* if $\overline{H_1 \cap H_2} = \overline{H_1} \cap \overline{H_2}$ [6].

The concept of the natural projection is associated to combine several DES into a single more complex DES. Given any event set Σ, let Σ_1 and Σ_2 be two event sets such that $\Sigma = \Sigma_1 \cup \Sigma_2$. Assume that they are not necessarily disjoint, i.e., $\Sigma_1 \cap \Sigma_2 \neq \phi$.

The *natural projection* p_i: $\Sigma^* \rightarrow \Sigma_i^*$ (i=1, 2) is defined by

$$p_i(\varepsilon) = \varepsilon$$

$$p_i(s\sigma) = \begin{cases} p_i(s)\sigma & \text{if } \sigma \in \Sigma_i \\ p_i(s) & \text{otherwise,} \end{cases}$$

for $s \in \Sigma^*$ and $\sigma \in \Sigma$. The natural projection p_i on a string s is to delete the element of σ of s which do not belong to Σ_i.

Let $\mathbf{G}_i := (Q_i, \Sigma_i, \delta_i, q_{i0}, Q_{i,m})$ for i = 1, 2. The *synchronous composition* of two automata is defined by $\mathbf{G}_1 \| \mathbf{G}_2 = Rch(Q_1 \times Q_2, \Sigma_1 \cup \Sigma_2, \delta_{12}, (q_{10}, q_{20}), Q_{1,m} \times Q_{2,m})$, according to

$$\delta_{12}((q_1,q_2),\sigma)=\begin{cases}(\delta_1(q_1,\sigma),\delta_2(q_2,\sigma)) & \text{if } \delta_1(q_1,\sigma) \text{ is defined and } \delta_2(q_2,\sigma) \text{ is defined}\\ (\delta_1(q_1,\sigma),q_2) & \text{if } \delta_1(q_1,\sigma) \text{ is defined and } \delta_2(q_2,\sigma) \text{ is not defined}\\ (q_1,\delta_2(q_2,\sigma)) & \text{if } \delta_1(q_1,\sigma) \text{ is not defined and } \delta_2(q_2,\sigma) \text{ is defined}\\ \text{Undefined} & \text{otherwise}\end{cases}$$

for $\sigma\in\Sigma_1\cup\Sigma_2$, $q_1\in Q_1$ and $q_2\in Q_2$. Note that *Rch* (…) means the reachable component of a DES. Using this, the closed and marked behaviors of synchronous composition can be obtained: $L(\mathbf{G}_1 \parallel \mathbf{G}_2) = p_1^{-1}(L(\mathbf{G}_1)) \cap p_2^{-1}(L(\mathbf{G}_2))$ and $L_m(\mathbf{G}_1 \parallel \mathbf{G}_2) = p_1^{-1}(L_m(\mathbf{G}_1)) \cap p_2^{-1}(L_m(\mathbf{G}_2))$, where p_i^{-1} is the inverse projection of p_i. It is known that $L(\mathbf{G}_1 \parallel \mathbf{G}_2) = L(\mathbf{G}_1) \parallel L(\mathbf{G}_2)$ and $L_m(\mathbf{G}_1 \parallel \mathbf{G}_2) = L_m(\mathbf{G}_1) \parallel L_m(\mathbf{G}_2)$. For the notational simplicity, we say, $L(\mathbf{G}_i)=L_i$. Basically, the two DES, $\mathbf{G}_1$ and $\mathbf{G}_2$, generate the larger DES $L_1 \| L_2$ cooperatively by synchronizing those events with common labels, while permitting events with different labels to occur whenever possible.

For a decentralized system, we assume that a centralized plant can be divided into several smaller decentralized plants as described in [18]. The concurrent operations of several decentralized plants could achieve the global objectives under certain conditions [7]. The framework of decentralized control is established as follows: let Σ_1, Σ_2, ···, Σ_n be the event alphabets of decentralized plants, $\mathbf{G}_1$, $\mathbf{G}_2$, ···, $\mathbf{G}_n$, respectively, and $\Sigma := \bigcup_{i=1}^n \Sigma_i$, where $\Sigma_i \cap \Sigma_j \neq \phi$, for $i, j \in \{1, 2, \cdots, n\}$ and $i \neq j$. It is assumed that $\Sigma_i = \Sigma_{ic} \cup \Sigma_{iu}$, and the control status of shared event of two subsystems agrees, i.e., $\Sigma_{iu} \cap \Sigma_j = \Sigma_i \cap \Sigma_{ju}$. Automatically, we have $\Sigma_c := \bigcup_{i=1}^n \Sigma_{ic}$, and $\Sigma_u := \bigcup_{i=1}^n \Sigma_{iu}$. The marked and closed behaviors of the centralized plant are, respectively

$$L_m = L_{1,m} \parallel L_{2,m} \parallel \cdots \parallel L_{n,m} = \bigcap_{i=1}^n (p_i)^{-1}(L_{i,m}),$$

$$L = L_1 \parallel L_2 \parallel \cdots \parallel L_n = \bigcap_{i=1}^n (p_i)^{-1}(L_i),$$

where $L_{i,m}$, $L_i \subseteq \Sigma^*$ represent respectively the marked and closed behaviors of decentralized plant $\mathbf{G}_i$. Decentralized specification can be established in a similar way: assume $E_i \subseteq L_{i,m}$ be $L_{i,m}$-closed language representing a specification on a decentralized plant $\mathbf{G}_i$. The specification for the centralized plant ($\mathbf{G}$) is a combination of all specifications applied to decentralized plants, that is, $E := \bigcap_{i=1}^n (p_i \mid_L)^{-1}(E_i)$, where $(p_i|_L)$ denotes the restriction of p_i on L: $(p_i \mid_L)^{-1}(E_i) = p_i^{-1}(E_i) \cap L$.

For a given decentralized specification (E_i), a decentralized supervisor ($\mathbf{S}_i$) can be obtained and the marked and closed behaviors of the closed-loop decentralized system ($\mathbf{S}_i/\mathbf{G}_i$) are $\kappa_{L_i}(E_i)$ and $\overline{\kappa_{L_i}(E_i)}$, respectively. A centralized supervisor $\mathbf{S}$ on the overall specification (E) can be synthesized and the marked and closed behaviors of the closed-loop system ($\mathbf{S}/\mathbf{G}$), are $\kappa_L(E)$ and $\overline{\kappa_L(E)}$, respectively. For the decentralized control, we are aiming to ensure the centralized supervisory control on the global plant should be the same as the concurrent actions of decentralized supervisory control applied in each local plant:

$$\bigcap_{i=1}^n (p_i \mid_L)^{-1}(\overline{\kappa_{L_i}(E_i)}) = \overline{\kappa_L(E)}$$

$$\text{or, } \overline{(\kappa_{L_1}(E_1))} \parallel \overline{(\kappa_{L_2}(E_2))} \parallel \cdots \parallel \overline{(\kappa_{L_n}(E_n))} = \overline{\kappa_{L_1 \parallel L_2 \parallel \cdots L_n}(E_1 \parallel E_2 \parallel \cdots \parallel E_n)}.$$

In a general case, this is not always true. Hence a set of conditions has developed to ensure the decentralized control will be as optimal as the centralized control [7, 10]. However the conditions developed in most researches are specification dependent which means, if a specification is changed (even minor rescheduling or recipe changes), the conditions should be verified again. The computational complexity would become definitely high if the specification changes are required frequently. Practically, in most of flexible chemical batch processes, this is the case since it is common to produce multi-products using multi-production sequences with the same equipment, especially in small and medium size companies.

Concept of structural decentralized supervisory control

To address such computational complexity issue, a different approach called *structural decentralized control system* has been proposed in Lee & Wong [11]. Basically, in this approach, once the structural conditions are verified to a given plant, the subsequent operations of decentralized control for a set of specifications will guarantee the same optimality as the centralized control action without further verification. Intuitively, it agrees with the fact that if the system structure is properly established, then the operation of these systems will be easier. These conditions are the first computationally efficient one that systematically guarantees the optimality of decentralized control in the structure of DES rather than on a certain specification only. This section introduces the framework of such structural decentralized control. We need the following definitions.

Definition 1.

Extension of nonconflicting condition defined in[6] to n languages: the languages $H_1, H_2, \ldots, H_n$ over the alphabet Σ are *nonconflicting* if

$$\overline{H_1 \cap H_2 \cap \cdots \cap H_n} = \overline{H_1} \cap \overline{H_2} \cap \cdots \cap \overline{H_n}.$$

Proposition 1.

Assume that$\{\kappa_L(E_i) | i = 1, 2, \ldots, n\}$ for $E_i \subseteq L$ ($i = 1, 2, \ldots, n$) are nonconflicting, where L is a closed language over Σ. Then we have $\kappa_L(\bigcap_{i=1}^n E_i) = \bigcap_{i=1}^n \kappa_{L_i}(E_i)$.

Refer Lee & Wong [11] for the proof.

Definition 2.

The marking process: Let H be a language over Σ and $\Sigma' \subseteq \Sigma$. It is said that H *marks* Σ' if $\Sigma^*\Sigma' \cap \overline{H} \subseteq H\Sigma'$.

That is, for any string $s \in \Sigma^*$ and any event $\sigma \in \Sigma'$, if $s\sigma \in \overline{H}$ and $s \in H$, then a language H is said to *mark* a given set of events Σ'.

Definition 3.

Mutual controllability condition: Let Σ_1, Σ_2, ··· , Σ_n be the event sets and $\Sigma := \bigcup_{i=1}^n \Sigma_i$, where $\Sigma_i \cap \Sigma_j \neq \phi$, for $i, j \in \{1, 2, \cdots, n\}$ and $i \neq j$. Assume $\Sigma_i = \Sigma_{ic} \cup \Sigma_{iu}$, and $\Sigma_{iu} \cap \Sigma_j = \Sigma_i \cap \Sigma_{ju}$. Let $H_i \subseteq \Sigma_i^*$. The languages H_i and H_j are *mutually controllable* if

$$\overline{H_i}(\Sigma_i \cap \Sigma_{ju}) \cap p_i^{ij}(p_j^{ij})^{-1}(\overline{H_j}) \subseteq \overline{H_i} \text{ and } \overline{H_j}(\Sigma_{iu} \cap \Sigma_j) \cap p_j^{ij}(p_i^{ij})^{-1}(\overline{H_i}) \subseteq \overline{H_j},$$

where p_i^{ij} and p_j^{ij} are the natural projections from $(\Sigma_i \cup \Sigma_j)^*$

to Σ_i^* and to Σ_j^*, respectively. Intuitively, we consider two plants are mutually controllable if one plant should be able to monitor all uncontrollable events (like breakdowns or power cuts) of the other plant. This would be crucial since in such cases, the corresponding plant needs to take necessary actions as required, to prevent blocking or other problems. From the synthesis process of the structural decentralized scheme, we notice that it is easy to satisfy the mutual controllability condition by adding selfloops to a local plant for any uncontrollable events that do not belong to that corresponding local plant. Note that fairly similar mutual controllability conditions are established in modular approaches in SCT with global specification languages [19].

We have the following proposition to establish the structural decentralized control in SCT.

Proposition 2.

Suppose that for $i, j \in \{1, 2, \cdots, n\}$ and $i \neq j$,

i) Shared Marking Condition: $L_{i,m}$ *and* $L_{j,m}$ *mark* $\Sigma_i \cap \Sigma_j$,

ii) Mutual controllability condition: L_i *and* L_j *are mutually controllable.*

Then for any $E_t \in \Phi_{L_{i,m}}$, *the set of* $L_{i,m}$ *- closed languages,*

$\bigcap_{i=1}^{n} (p_i|_L)^{-1}(\overline{\kappa_{L_i}(E_i)}) = \overline{\kappa_L(E)}$, *and*

$p_i(\overline{\kappa_L(E)})$ *is not blocking with respect to* $L_{i,m}$.

For the details, refer to Lee & Wong [11]. This proposition states that if the marked behaviors of any two subsystems mark their shared events and any two subsystems are mutually controllable, then decentralized syntheses and concurrent control of decentralized supervisors for any $L_{i,m}$ -closed specifications will guarantee the optimality compared to the centralized synthesis.

Main result

This section presents the problem statement and the main results of this paper. In the coordination scheme, we assume that the structural decentralized control is established among local plants. Then the coordination plant consisting of synchronized events is established using natural projections. The specification, which is an $L_{h,m}$-closed language, given in the coordination plant can be established, where $L_{h,m}$ is the marked behavior of the coordination plant. Finally the coordinator can be synthesized. The structural conditions for the existence of such a coordinator ensuring the global optimality are developed in this paper. Note that the process to synthesize or construct such a coordination scheme is similar to that of the original structural decentralized control scheme, which will be published separately.

For the formulation of the problem in this paper, we define the coordinator as follows (see Figure 1 for the scheme). Let Σ_1, Σ_2, $\cdots$, Σ_n be the event alphabets of decentralized plants, $\mathbf{G}_1$, $\mathbf{G}_2$, $\cdots$, $\mathbf{G}_n$, respectively, and $\Sigma := \bigcup_{i=1}^{n} \Sigma_i$, where $\Sigma_i \cap \Sigma_j \neq \phi$, for $i, j \in \{1, 2, \cdots, n\}$ and $i \neq j$. It is also assumed that $\Sigma_i = \Sigma_{ic} \cup \Sigma_{iu}$ and $\Sigma_{iu} \cap \Sigma_j = \Sigma_i \cap \Sigma_{ju}$. Let L_i, $L_{i,m} \subseteq \Sigma^*$ be respectively the closed and marked behaviors of decentralized plant $\mathbf{G}_i$. Also assume $L_i = \overline{L_{i,m}}$. We define a coordinator only with the synchronized shared events. Let the event set in the coordinator be $\Sigma_h = \bigcup_{i=1}^{n}(\Sigma_i \cap (\bigcup_{j=1}^{n}(\Sigma_j))$, $j \neq i$ and p_h be the natural projection from Σ^* to Σ_h^*. Consider $L_h = p_h(L)$ and $L_{h,m} = p_h(L_m)$ as the closed and marked behaviors of the coordination plant, respectively. Let the specification on the coordination system $E_h \subseteq L_{h,m}$ be an $L_{h,m}$-closed language representing the interactions among local plants. The coordination supervisor $\mathbf{S}_h$ for the specification E_h on the plant $\mathbf{G}_h$ can be synthesized. The marked and closed behaviors of the closed-loop system $\mathbf{S}_h/\mathbf{G}_h$ are $\kappa_{L_h}(E_h)$ and $\overline{\kappa_{L_h}(E_h)}$, respectively. The equivalent global specification with the specification given in the coordination plant E_h is

$$E' := \bigcap_{i=1}^{n}(p_i|_L)^{-1}(E_i) \cap (p_h|_L)^{-1}(E_h) = E \cap (p_h|_L)^{-1}(E_h).$$

Definition 4.

Coordinator: Assume the decentralized plants and the coordination plant to be defined as above. Then we define $\mathbf{S}_h$ as a coordinator if

$$\overline{\kappa_L(E')} = \bigcap_{i=1}^{n}(p_i|_L)^{-1}(\overline{\kappa_{L_i}(E_i)}) \cap (p_h|_L)^{-1}(\overline{\kappa_{L_h}(E_h)}). \quad (1)$$

That is, a coordinator is a coordination supervisor $\mathbf{S}_h$ on $\mathbf{G}_h$ which guarantees the concurrent decentralized supervisions combined with it to achieve the optimal centralized control objective. The conditions for the existence of such coordinator are established as follows.

Problem 1.

For all pairs of the systems $\mathbf{G}_1$, $\mathbf{G}_2$, *…,* $\mathbf{G}_n$, *and* $\mathbf{G}_h$ *defined as above and for any* $L_{h,m}$ *-closed language* E_h, *under what condition is it true that the supervisor* $\mathbf{S}_h$ *for* E_h *on* $\mathbf{G}_h$ *is a coordinator, i.e., Eq. (1) is true?*

Thus, Problem 1 is to find the conditions under which a coordination plant $\mathbf{G}_h$ and decentralized syntheses and control of $\mathbf{G}_1$, $\mathbf{G}_2$, …, $\mathbf{G}_n$, do not result in the loss of optimality compared to the centralized synthesis.

The sufficient conditions for the existence of the coordinator for Problem 1 are developed in this paper:

Theorem 1.

Conditions for the existence of a coordinator: Let Σ, Σ_i, Σ_{ic}, Σ_{iu}, Σ_h, $L_{i,m}$, L_i *(i = 1,2, …, n),* L_h, $L_{h,m}$, L_m, *and L be given as above. Suppose*

i) Shared event marking condition: for all pairs of the systems $\mathbf{G}_1$, $\mathbf{G}_2$, … *and* $\mathbf{G}_n$ *(i, j =* 1,2, *…, n and* $i \neq j$*),* $L_{i,m}$ *and* $L_{j,m}$ *mark* $\Sigma_i \cap \Sigma_j$,

ii) Mutually controllability condition: for all pairs of the systems $\mathbf{G}_1$, $\mathbf{G}_2$, …, $\mathbf{G}_n$ *and* $\mathbf{G}_h$, L_i *and* L_j *(including* L_h*) are mutually controllable.*

Then Problem 1 is solved, that is, a supervisor for a $L_{h,m}$*-closed sublanguage is a coordinator.*

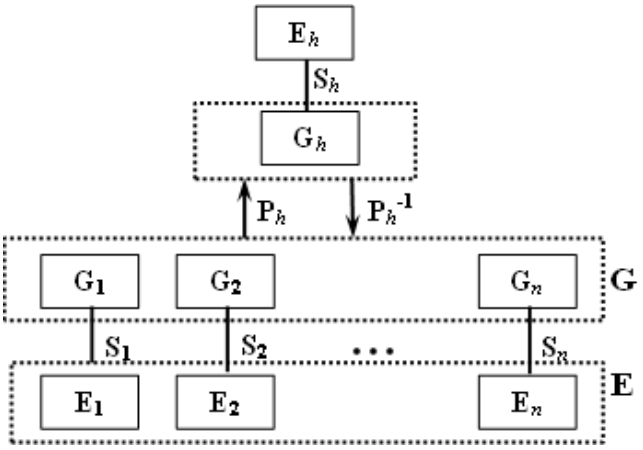


Figure 1 Coordinating scheme

Note that Theorem 1 is an extension of Proposition 2 into the coordination scheme of the structural decentralized control. It is worth to mention that the shared event marking condition does not include $\mathbf{G}_h$ in the statement while the mutual controllability condition does. Since the coordination plant should also need to monitor any uncontrollable events occurring in any local plants to prevent blocking, it is included in the mutual controllability condition. Hence it is only necessary to prove whether the shared event marking condition is extended to the coordination plant since the proof of mutual controllability condition is trivial. Since the conditions are still dependent on the system structure, unlike the researches in [14, 15], once the conditions are verified for a given coordination system structure, a set of specifications can automatically satisfy the conditions and hence the corresponding coordinator can be synthesized without any further verification. This would definitely provide significant computational savings especially when there are frequent minor changes of specifications.

The proof of Theorem 1 is as follows: under the assumptions given in (*i*) of Theorem 1, we need to prove whether the pair of the systems ($\mathbf{G}_i$, $\mathbf{G}_h$), for i=1, 2, … n, automatically satisfies the shared-event-marking condition. Proposition 3 provides it.

Proposition 3.
Let Σ_i, Σ_h, $L_{i,m}$, $L_{h,m}$ *be defined as in the above, for* i=1 ,2, ... , n. *Then* $L_{i,m}$ *and* $L_{h,m}$ *mark* $\Sigma_i \cap \Sigma_h$.
The following lemmas need to be established to prove Proposition 3.

Lemma 1.
Let L_i, $L_{i,m}$ *and* p_h *be defined as above where* i=1, 2,... , n. *Then* $p_h(L_i)=p_h(L_{i,m})$.

Proof for Lemma 1:
Since it is always $p_h(L_i) \supseteq p_h(L_{i,m})$, we show the other inclusion only. Let $s \in p_h(L_i)$. Then there exists a string $u \in L_i$ such that $s=p_h(u)$. Since $L_i = \overline{L_{i,m}}$, there exists a string $v \in \Sigma_i^*$ such that $uv \in L_{i,m}$. Hence $p_h(uv) \in p_h(L_{i,m})$. We consider two cases: Firstly, if $v \in (\Sigma_i - \Sigma_h)^*$, then $p_h(uv)=p_h(u)p_h(v)=p_h(u)=s \in p_h(L_{i,m})$. In other case, which is $v \notin (\Sigma_i - \Sigma_h)^*$, there are events $\sigma_1,\sigma_2,... ,\sigma_n \in \Sigma_h$ and $w_1,w_2,... ,w_n,w_{n+1} \in (\Sigma_i - \Sigma_h)^*$ such that $v=w_1\sigma_1 w_2\sigma_2 ... w_n\sigma_n w_{n+1}$. Since $uv \in L_{i,m}$, $uv=uw_1\sigma_1 w_2\sigma_2...w_n\sigma_n w_{n+1} \in L_{i,m}$. Hence $uw_1\sigma_1 \in \overline{L_{i,m}}$. Also, since $u \in L_i$ and $w_1 \in (\Sigma_i - \Sigma_h)^*$, $uw_1 \in \Sigma_i^*$. Since $\sigma_1 \in \Sigma_h$, $\sigma_1 \in \Sigma_i \cap \Sigma_j$ for some j. By the assumption of the condition (*i*) in Theorem 1 between two systems $\mathbf{G}_i$ and $\mathbf{G}_j$, $uw_1 \in L_{i,m}$. Therefore,
$P_h(uw_1)=p_h(u)p_h(w_1)=p_h(u)=s \in p_h(L_{i,m})$.
The following Lemma 2 and 3 are used for Lemma 4.

Lemma 2
For alphabets Σ_o, Σ_1, Σ_2 *with* $\Sigma=\Sigma_1 \cup \Sigma_2$ *and* $\Sigma_o \subseteq \Sigma$, *let* $L_1 \subseteq \Sigma_1^*$, $L_2 \subseteq \Sigma_2^*$, *and let* $p_o:\Sigma^* \to \Sigma_o^*$ *be the natural projection. Then* $p_o(L_1 \| L_2) \subseteq (p_o L_1) \| (p_o L_2)$.

Proof for Lemma 2:
Since $L_1 \| L_2 = p_1^{-1}(L_1) \cap p_2^{-1}(L_2)$, where p_i: $\Sigma^* \to \Sigma_i^*$ and $(p_o L_1) \| (p_o L_2) = (p_1^o)^{-1}(p_o L_1) \cap (p_2^o)^{-1}(p_o L_2)$,
where $p_i^o : \Sigma_o^* \to (\Sigma_o \cap \Sigma_i)^*$, to prove the above equation, consider a string $s \in p_o(L_1 \| L_2)$. Then we have a string $u \in L_1 \| L_2$ such that $s=p_o(u)$. Hence $u \in p_1^{-1}(L_1) \cap p_2^{-1}(L_2)$ and $s=p_o(u)$. So, $p_1(u) \in L_1$ and $p_2(u) \in L_2$. Therefore, $p_o p_1(u) \in p_o(L_1)$ and $p_o p_2(u) \in p_o(L_2)$. Also $p_i^o(s) = p_i^o(p_o(u)) = p_i(p_o(u)) = p_o(p_i(u))$. Hence, $p_1^o(s) \in p_o(L_1)$ and $p_2^o(s) \in p_o(L_2)$. So, $s \in (p_1^o)^{-1}(p_o L_1) \cap (p_2^o)^{-1}(p_o L_2) = (p_o L_1) \| (p_o L_2)$.

Lemma 3.
Let Σ_o, Σ_1, Σ_2, p_o, p_i, p_i^o *be defined as in Lemma 2. Suppose that* $\Sigma_1 \cap \Sigma_2 \subseteq \Sigma_o$. *Then* $p_o(L_1 \| L_2) = (p_o L_1) \| (p_o L_2)$.

Proof for Lemma 3:
One inclusion ($\subseteq$) is already shown in Lemma 2. For the other inclusion, we firstly show the following claims.

Claim 1.
For $L \subseteq \Sigma_i^*$ (for i=1,2), it is always $(p_i^o)^{-1}(p_o L) = p_o(p_i)^{-1}(L)$.

Proof of Claim 1:
We show the case i=1. For one inclusion ($\supseteq$), let $s \in p_o(p_1)^{-1}(L)$. Then there exists a string $u \in (p_1)^{-1}(L)$ such that $s=p_o(u)$. Hence $p_1(u) \in L$. So $p_o(p_1(u)) \in p_o(L)$. Note that $p_1^o(s) = p_1^o(p_o(u)) = p_1(p_o(u)) = p_o(p_1(u)) \in p_o(L)$. Hence $s \in (p_1^o)^{-1} p_o(L)$. For the reverse inclusion ($\subseteq$), let $s \in (p_1^o)^{-1} p_o(L) \in \Sigma_o^*$. Hence $p_1^o(s) \in p_o(L)$. We have two cases: Firstly, if $s \in (\Sigma_o - \Sigma_1)^*$, then $p_1^o(s) = \varepsilon \in p_o(L)$. So, there exists a string $u \in L$ such that $p_o(u)=\varepsilon$, the empty string. Hence $u \in (\Sigma_1 - \Sigma_o)^*$. Since $p_1(su)=u \in L$, one has $su \in p_1^{-1}(L)$. Hence $p_o(su) = s \in p_o p_1^{-1}(L)$. In the other case, if $s \notin (\Sigma_o - \Sigma_1)^*$, one can consider a string $s=w_1\sigma_1 w_2\sigma_2...w_m\sigma_m w_{m+1}$, where $\sigma_1,\sigma_2,... ,\sigma_m \in (\Sigma_1 \cap \Sigma_o)$, and $w_1,w_2,... ,w_m,w_{m+1} \in (\Sigma_o - \Sigma_1)^*$. Since

$p_1^o(s) \in p_o(L)$, one has that $p_1^o(s) = \sigma_1\sigma_2..\sigma_m \in p_o(L)$. Hence there exists a string $u \in L$ such that $p_1^o(s) = \sigma_1\sigma_2..\sigma_m \in p_o(u)$. Since $u \in L \subseteq \Sigma_1^*$, one has that $u=u_1\sigma_1u_2\sigma_2...u_m\sigma_mu_{m+1}$, where $u_1,u_2,...,u_m,u_{m+1} \in (\Sigma_1-\Sigma_o)^*$. Define a string $v:= w_1u_1\sigma_1w_2u_2\sigma_2...w_mu_m\sigma_mw_{m+1}u_{m+1}$. Then $p_1(v)=u_1\sigma_1u_2\sigma_2...u_m\sigma_mu_{m+1}=u \in L$. Hence $v \in (p_1)^{-1}(L)$. Thus $p_o(v) \in p_o(p_1)^{-1}(L)$. Also $p_o(v)= w_1\sigma_1w_2\sigma_2...w_m\sigma_mw_{m+1} = s$. Therefore $s = p_o(v) \in p_o(p_1)^{-1}(L)$. This proves Claim 1.

Claim 2.
Suppose that $\Sigma_1 \cap \Sigma_2 \subseteq \Sigma_o$, *Then* $p_o p_1^{-1}(L_1) \cap p_o p_2^{-1}(L_2) = p_o(p_1^{-1}(L_1) \cap p_2^{-1}(L_2))$.

Proof of Claim 2:
Let $s \in p_o p_1^{-1}(L_1) \cap p_o p_2^{-1}(L_2)$. Then $s \in p_o p_1^{-1}(L_1)$ and $s \in p_o p_2^{-1}(L_2)$. Hence there exist a string $u \in p_1^{-1}(L_1)$ such that $s=p_o(u)$. So $p_1(u) \in L_1$. Also there is a string $v \in p_2^{-1}(L_2)$ such that $s=p_o(v)$. So $p_2(v) \in L_2$ and $s=p_o(v)$. If $s=\varepsilon$, then $u,v \in (\Sigma-\Sigma_o)^*$. Since $\Sigma_1 \cap \Sigma_2 \subseteq \Sigma_o$, one has that $\Sigma-\Sigma_o \subseteq \Sigma- (\Sigma_1 \cap \Sigma_2)=(\Sigma_1-\Sigma_2) \cap (\Sigma_2-\Sigma_1)$. So, $u,v \in ((\Sigma_1-\Sigma_2) \cap (\Sigma_2-\Sigma_1))^*$. Therefore $p_1(u) \in (\Sigma_1-\Sigma_2)^*$ and $p_2(v) \in (\Sigma_2-\Sigma_1)^*$. Let $w:=p_1(u)p_2(v)$. Then $p_1(w)= p_1(u) \in L_1$ and $p_2(w)=p_2(v) \in L_2$. Hence $w \in p_1^{-1}(L_1) \cap p_2^{-1}(L_2)$. So $p_o(w)=p_o(p_1(u)p_2(v))$ $=p_op_1(u)p_op_2(v)=p_1p_o(u)p_2p_o(v)=p_1(\varepsilon)p_2(\varepsilon)=\varepsilon=s$. Hence $s \in p_o(p_1^{-1}(L_1) \cap p_2^{-1}(L_2))$. If $s \neq \varepsilon$, then $s=\sigma_1\sigma_2...\sigma_n$, for some $n \geq 1$ and $\sigma_i \in \Sigma_o$. We can have $u=u_1\sigma_1u_2\sigma_2...u_n\sigma_n u_{n+1}$ and $v=v_1\sigma_1 v_2\sigma_2...v_n\sigma_nv_{n+1}$, for some $u_i,v_i \in (\Sigma-\Sigma_o)^*$. Since $\Sigma_1 \cap \Sigma_2 \subseteq \Sigma_o$, one has that $\Sigma-\Sigma_o \subseteq (\Sigma_1-\Sigma_2) \cap (\Sigma_2-\Sigma_1)$. Therefore $p_1(u_i) \in (\Sigma_1-\Sigma_2)^*$ and $p_2(v_i) \in (\Sigma_2-\Sigma_1)^*$. Define a string

$$w := p_1(u_1)p_2(v_1)\begin{Bmatrix} \sigma_1 & \text{if } \sigma_1 \in (\Sigma_1 \cap \Sigma_2) \\ p_1(\sigma_1)p_2(\sigma_1) & \text{if } \sigma_1 \in \Sigma-(\Sigma_1 \cap \Sigma_2) \end{Bmatrix} p_1(u_2)p_2(v_2)...$$
$$p_1(u_n)p_2(v_n)\begin{Bmatrix} \sigma_n & \text{if } \sigma_n \in (\Sigma_1 \cap \Sigma_2) \\ p_1(\sigma_n)p_2(\sigma_n) & \text{if } \sigma_n \in \Sigma-(\Sigma_1 \cap \Sigma_2) \end{Bmatrix} p_1(u_{n+1})p_2(v_{n+1}).$$

So,
$p_1(w) = p_1(u_1)p_1(\sigma_1)p_1(u_2)p_1(\sigma_2)...p_1(u_n)p_1(\sigma_n)p_1(u_{n+1}) \in L_1$
$p_2(w) = p_2(v_1)p_2(\sigma_1)p_2(v_2)p_2(\sigma_2)...p_2(v_n)p_2(\sigma_n)p_2(v_{n+1}) \in L_2$.
Hence $w \in p_1^{-1}(L_1) \cap p_2^{-1}(L_2)$. So $p_o(w)=\sigma_1\sigma_2...\sigma_n=s$. Therefore $s \in p_o(p_1^{-1}(L_1) \cap p_2^{-1}(L_2))$.
We now can prove Lemma 3.

Proof of Lemma 3:
We now show $p_o(L_1 \parallel L_2) \supseteq (p_oL_1) \parallel (p_oL_2)$.

$$\begin{aligned}(p_oL_1) \parallel (p_oL_2) &= (p_1^o)^{-1}p_o(L_1) \cap (p_2^o)^{-1}p_o(L_2) \\ &= p_o p_1^{-1}(L_1) \cap p_o p_2^{-1}(L_2) \quad \text{(By Claim 1)} \\ &\subseteq p_o(p_1^{-1}(L_1) \cap p_2^{-1}(L_2)) \quad \text{(By Claim 2)} \\ &\subseteq p_o(L_1 \parallel L_2).\end{aligned}$$

Lemma 4.
Let Σ_i *be the event alphabets for the plant* $\mathbf{G}_i$ *for* $i=1, 2,\cdots, n$. *Assume* $\Sigma := \bigcup_{i=1}^n \Sigma_i$. *and* $\Sigma_i \cap \Sigma_j \neq \phi$, *for* $i, j \in \{1, 2,\cdots, n\}$ *and* $i \neq j$. *Let* $\Sigma_h := \bigcup_{i \neq j}(\Sigma_i \cap \Sigma_j)$, *and let* p_i *and* p_h *be the natural projection from* Σ^* *to* Σ_i^* *and to* Σ_h^*, *respectively. Then for* $L_i \subseteq \Sigma_i^*$

$$p_h(L_1 \parallel L_2 \parallel ... \parallel L_n) = p_h(L_1) \parallel p_h(L_2) \parallel ... \parallel p_h(L_n).$$

Proof of Lemma 4:
We have

$$\begin{aligned}p_h(L_1 \parallel L_2 \parallel ... \parallel L_n) &= p_h(L_1) \parallel p_h(L_2 \parallel ... \parallel L_n) \quad (\text{since } \Sigma_1 \cap (\textstyle\bigcup_{j\neq 1}\Sigma_j) \subseteq \Sigma_h \text{ and by Lemma 3}) \\ &= p_h(L_1) \parallel p_h(L_2) \parallel p_h(L_3 \parallel ... \parallel L_n) \quad \text{(by Lemma 3)} \\ &\vdots \\ &= p_h(L_1) \parallel p_h(L_2) \parallel ... \parallel p_h(L_{n-1} \parallel L_n) \quad \text{(by Lemma 3)} \\ &= p_h(L_1) \parallel p_h(L_2) \parallel ... \parallel p_h(L_{n-1}) \parallel p_h(L_n) \quad \text{(by Lemma 3)}\end{aligned}$$

The following lemma is established from Lemma 1 and Lemma 4

Lemma 5.
Let $L_{h,m}=p_h(L_m)$ *and* $L_h=p_h(L)$. *Then* $L_h=L_{h,m}$.

Proof of Lemma 5:
It can be shown in turn

$$\begin{aligned}L_h &= p_h(L_1 \parallel L_2 \parallel ... \parallel L_n) \\ &= p_h(L_1) \parallel p_h(L_2) \parallel ... \parallel p_h(L_{n-1}) \parallel p_h(L_n) \quad \text{(by Lemma 4)} \\ &= p_h(L_{1,m}) \parallel p_h(L_{2,m}) \parallel ... \parallel p_h(L_{n,m}) \quad \text{(by Lemma 1)} \\ &= p_h(L_{1,m} \parallel L_{2,m} \parallel ... \parallel L_{n,m}) \quad \text{(by Lemma 4)} \\ &= L_{h,m}\end{aligned}$$

Now we can prove Proposition 3.

Proof of Proposition 3:
Since we assume that $L_{i,m}$ for $i=1,2,...,n$, marks its shared events, $L_{i,m}$ marks $\Sigma_i \cap \Sigma_h$. and hence $L_{h,m}$ also marks $\Sigma_i \cap \Sigma_h$ (Lemma 5).
The proposition 3 shows that some of the structural properties of the given subsystems $\mathbf{G}_i$ are inherited by the coordination plant $\mathbf{G}_h$.

Proof of Theorem 1:
Theorem 1 is proved by Proposition 2 and Proposition 3.
Note that Theorem 1 inherits some of the structural decentralized control properties of Proposition 2. Note that in general, the coordination plant may not be mutually controllable with all local plants. However, from the synthesis algorithm we have developed, it is found that only selfloops of uncontrollable events are necessary in the coordination plant (indicating the monitoring of those events). Fundamentally, the coordination plant allows all uncontrollable events in local plants to happen but requiring necessary monitoring actions for the coordination. It seems that this condition is too strict; however, this is because the conditions are applied to a structure of the system. Definitely, this condition can be relaxed if the conditions become specification dependent as shown in [15]. The issue on how to relax such a strict condition while maintaining the structural condition property is still an open question.

Example
This section presents an example to illustrate the results. The software package CTCT developed by Wonham [17] is used

to carry out all computations presented in this section. Consider a chemical batch reactor shown in Figure 2. The list of components in the chemical reactor is shown in Table 1. Note that the duration of the chemical reaction is timed by the timer T_1, which can be set for 10 or 15 minutes. A 30-second timer T_2 is for the timing requirement to add catalysts. The chemical reaction is processed with any combination of two materials from three available materials, A, B and C, and one catalyst either D or E. The final products will be drained out using one of three drain exits, F, G or H. Figure 3 shows the different processes for the production. To represent a certain combination of materials, a catalyst or a drain exit used, we introduced the controllable shared event σ_1, σ_2, ..., σ_{10} to indicate the beginning of a certain process to enforce that the processes are operating serially. For example, the shared event σ_1 is for indicating a mixture of materials A and B. The other shared events have similar roles.

Note that without any control actions applied, each component can work independently and asynchronously. Since the number of the elementary components is 20 (as shown in Table 1) and if each component is assumed to have 2 states each, the total number of states could be more than 1.0×10^6. This is too big to analyze as a whole. Hence a decentralized control of DES is an ideal choice for the study. From the analysis of operations, the whole plant is decomposed into three subplants as shown in Figure 2: the filling subplant ($\mathbf{G}_1$), the reaction subplant ($\mathbf{G}_2$) and the draining subplant ($\mathbf{G}_3$).

Table 1 Component list of the chemical reactor.

Component	Labels
A reaction tank	No labels
Three material feed valves	V_1, V_2, V_3
Three material feed pumps	P_1, P_2, P_3
Three drain valves	V_4, V_5, V_6
Three drain pumps	P_4, P_5, P_6
Two supply valves for catalysts	V_7, V_8
Low/high level sensors for the tank	WL_1, WH_1
Heater with a temperature probe	TP_1
Continuous reaction controller	C_1
Two timers	T_1, T_2

Note that there are no shared components among subplants. Due to the nature of this chemical batch plants, the operation procedures are required to be modified frequently to meet the demands for production flexibility. Therefore, the structural decentralized DES framework is an ideal choice to deal with the computational complexity issue.

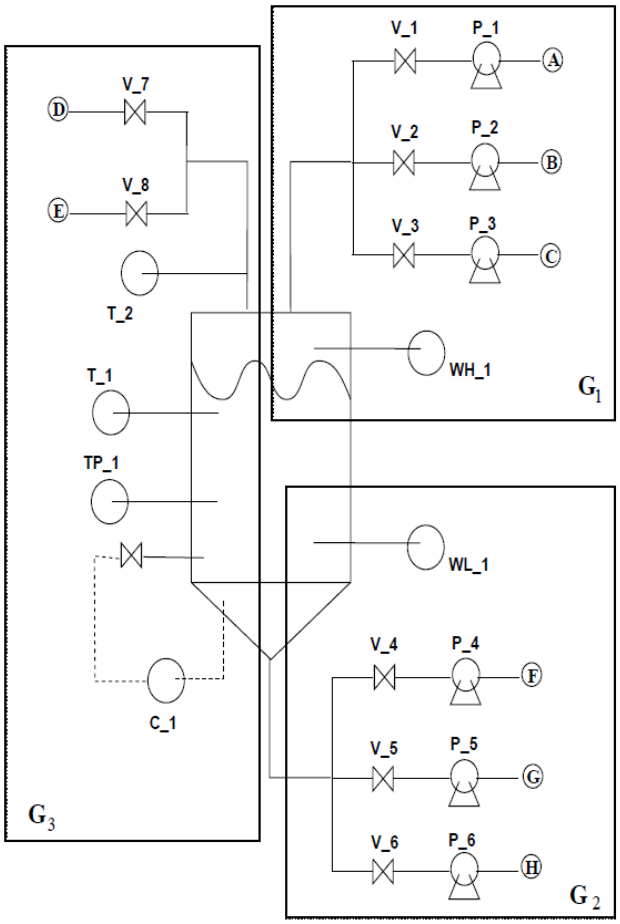


Figure 2 Batch reactor diagram

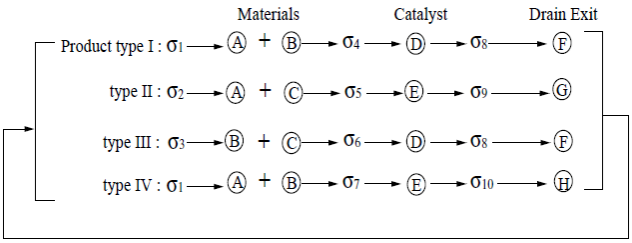


Figure 3 Process for the batch reaction Process

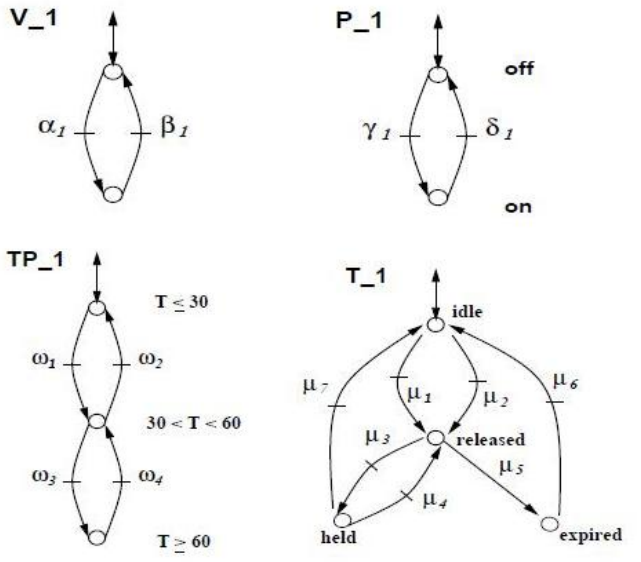


Figure 4 DES models for some elementary components of the plant

To synthesize a supervisor, we need to create DES models for plants and specifications. Most components have only two states: on and off for pumps, open and closed for valves, or similar to others, with exceptions of the temperature probe TP_1 with 3 states and the timer T_1 with 4 states. Figure 4 shows the DES automata model for some typical components. In the figure, a state is represented by a circle and an event is represented by an arrow (with a label) from an exit state to the entrance state. Also an entering arrow ($\rightarrow\circ$) indicates the initial state and an exiting arrow ($\leftarrow\circ$) represents the marker state, while the double arrow ($\leftrightarrow\circ$) shows the initial state and the marker state. The arrow with a bar in the middle represents a controllable event. Table 2 shows the components in each subplant with their event labels.

Table 2 Components with event labels of decentralized plants

Plant	Labels	Components with event labels
Filling subplant	$\mathbf{G}_1$	$V_1(\alpha_1, \beta_1)$, $V_2(\alpha_2, \beta_2)$, $V_3(\alpha_3, \beta_3)$, $P_1(\gamma_1, \delta_1)$, $P_2(\gamma_2, \delta_2)$, $P_3(\gamma_3, \delta_3)$,$WH_1(\eta_1,\eta_2)$
Reaction subplant	$\mathbf{G}_2$	$V_7(\alpha_7, \beta_7)$, $V_8(\alpha_8, \beta_8)$, $TP_1(\omega_1,\omega_2,\omega_3,\omega_4)$, $C_1(\xi_1,\xi_2)$, $T_1(\mu_8,\mu_9)$, $T_2(\mu_1,\mu_2,\mu_3,\mu_4,\mu_5,\mu_6,\mu_7)$
Draining subplant	$\mathbf{G}_3$	$V_4(\alpha_4, \beta_4)$, $V_5(\alpha_5, \beta_5)$, $V_6(\alpha_6, \beta_6)$, $P_4(\gamma_4, \delta_4)$, $P_5(\gamma_5, \delta_5)$, $P_6(\gamma_6, \delta_6)$,$WL_1(\eta_3,\eta_4)$

To ensure the sequential operations of these three subplants, three additional controllable shared events (λ_1,λ_2,λ_3) are introduced to indicate a completion of the filling process, the reaction process and the draining process (this also means the completion of one cycle of the whole process), respectively. These should be a part of DES models of each subplant. Figure 5 represents such a DES model for $\mathbf{G}_1$ representing that after the shared events (σ_1, σ_2, or σ_3), the feed valves are allowed to be closed (β_1, β_2 and β_3). Then with the occurrences of the events λ_1 and λ_3, a cycle of the operation is completed. Similar DES models for $\mathbf{G}_2$ & $\mathbf{G}_3$ can also be established.

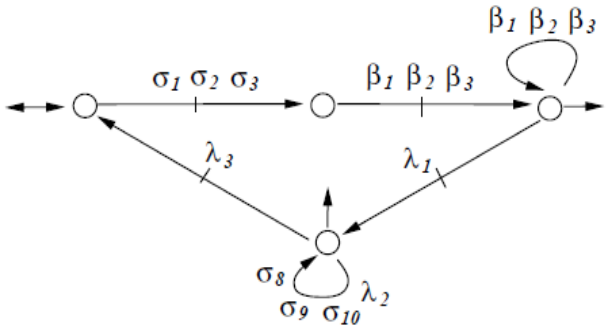


Figure 5 DES models for the synchronization shared events for $\mathbf{G}_1$

The DES model for each subplant is obtained by the synchronous composition of the elementary components and a DES shown in Figure 5 (or similar). The number of states in the decentralized synthesis is much smaller than the centralized one as the number of components in each decentralized plant is much smaller than the centralized one. The event sets for the decentralized plants are as follows

$\Sigma_1=\{\alpha_1, \alpha_2, \alpha_3, \beta_1, \beta_2, \beta_3, \gamma_1, \gamma_2, \gamma_3, \delta_1, \delta_2, \delta_3, \eta_1, \eta_2, \sigma_1, \sigma_2, \sigma_3, \sigma_8, \sigma_9, \sigma_{10}, \lambda_1, \lambda_2, \lambda_3\}$,

$\Sigma_2=\{\alpha_7, \alpha_8, \beta_7, \beta_8, \xi_1, \xi_2, \omega_1, \omega_2, \omega_3, \omega_4, \mu_1, \mu_2, \mu_3, \mu_4, \mu_5, \mu_6, \mu_7, \mu_8, \mu_9, \sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6, \sigma_7, \lambda_1, \lambda_2, \lambda_3\}$

$\Sigma_3=\{\alpha_4, \alpha_5, \alpha_6, \beta_4, \beta_5, \beta_6, \gamma_4, \gamma_5, \gamma_6, \delta_4, \delta_5, \delta_6, \eta_3, \eta_4, \sigma_4, \sigma_5, \sigma_6, \sigma_7, \sigma_8, \sigma_9, \sigma_{10}, \lambda_1, \lambda_2, \lambda_3\}$.

The shared event set is $\Sigma_h=\{\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6, \sigma_7, \sigma_8, \sigma_9, \sigma_{10}, \lambda_1, \lambda_2, \lambda_3\}$.

Now we establish a DES model for the specifications for each plant in the local level. We will show in here for the plant $\mathbf{G}_1$. It is assumed that the specification for the plant $\mathbf{G}_1$, denoted by E_1, is as follows (see Figure 6 for the DES model of E_1):

1. For the mixture of the materials A and B (σ_1), open the valves V_1 and V_2 (α_1, α_2) and turn on the pumps P_1 and P_2 (γ_1,γ_2).
2. For the mixture of the materials B and C (σ_2), open the valves V_2 and V_3 (α_2, α_3) and turn on the pumps P_2 and P_3 (γ_2,γ_3).
3. For the mixture of the materials A and C (σ_3), open the valves V_1 and V_3 (α_1, α_3) and turn on the pumps P_1 and P_3 (γ_1,γ_3).
4. When the level in the tank reaches $L \geq 100l$ (η_1), turn off the pumps (δ_1,δ_2,δ_3) and close the valves (β_1, β_2, β_3) whichever necessary. Signal λ_1 to the other plants.

5. The selfloops at the initial state mean that the subplant $\mathbf{G}_1$ does not restrict the behaviors of the other subplants.

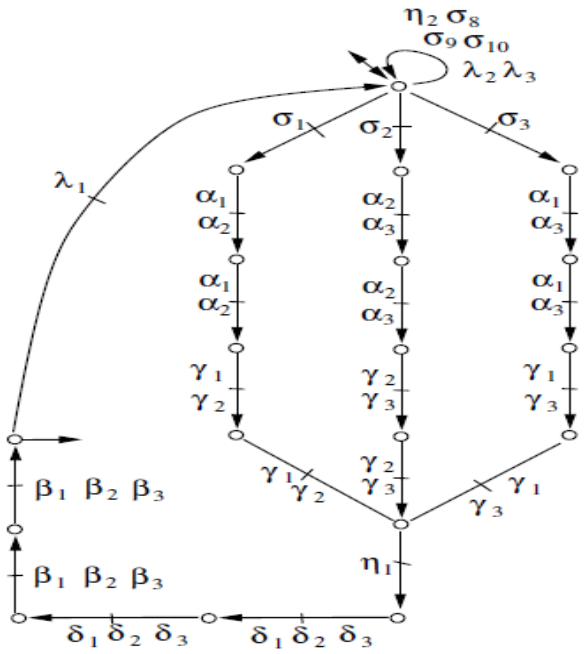


Figure 6 DES models for the specification E_1 for $\mathbf{G}_1$

The DES model of specifications for the other plants, E_2 and E_3, can be obtained in a similar way and given in Figure 7 and 8, respectively.

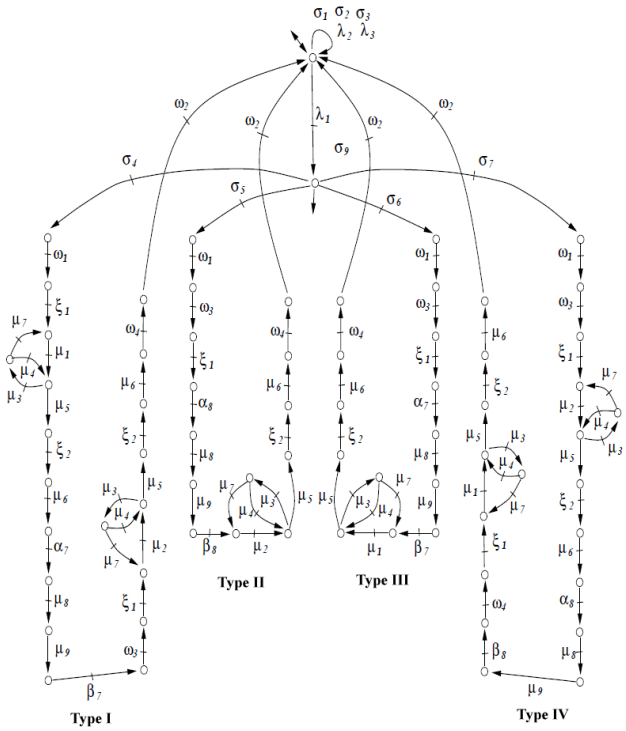


Figure 7 DES models for the specification E_2 for $\mathbf{G}_2$

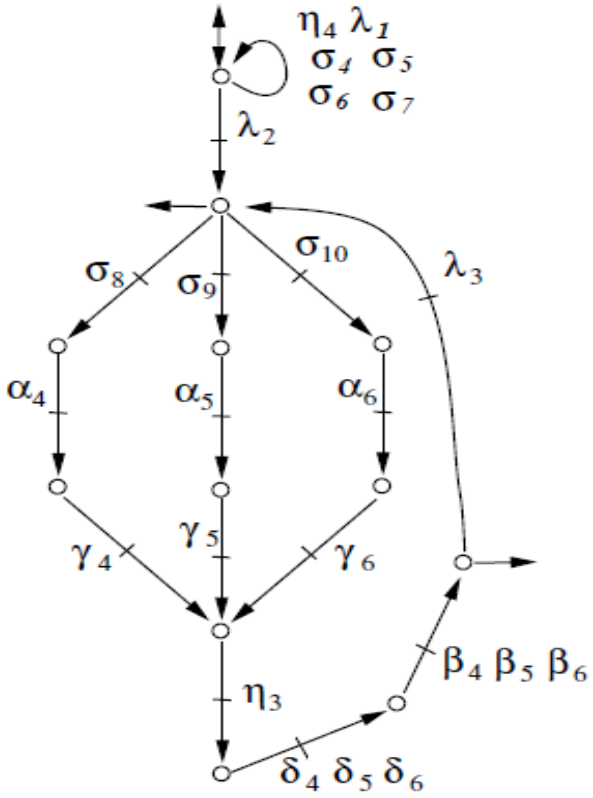


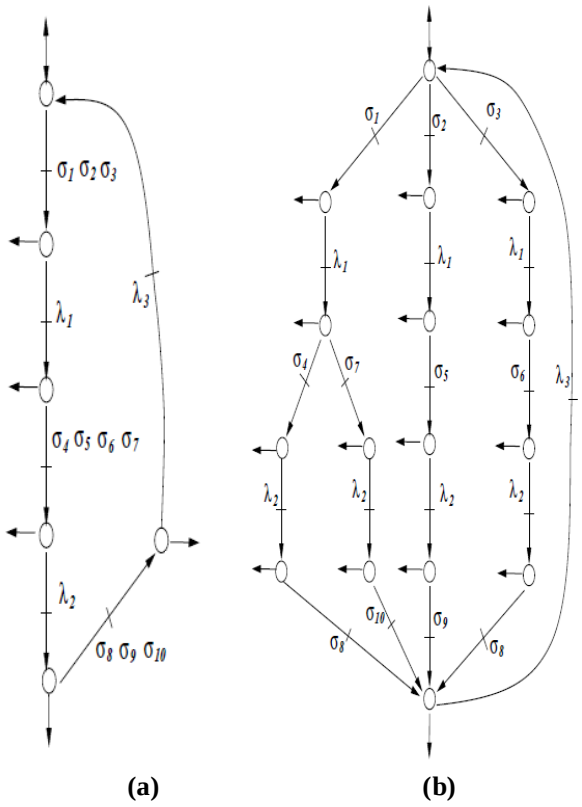
Figure 8 DES models for the specification E_3 for $\mathbf{G}_3$

Then the decentralized supervisors $\mathbf{S}_1$ for E_1, $\mathbf{S}_2$ for E_2, and $\mathbf{S}_3$ for E_3 in the decentralized plants $\mathbf{G}_1$, $\mathbf{G}_2$ and $\mathbf{G}_3$, are obtained by the supremal controllable sublanguage concept [4]. The conditions for the structural decentralized control in Proposition 2 need to be verified for the structure of this example ($\mathbf{G}_1$, $\mathbf{G}_2$, and $\mathbf{G}_3$) before the coordination plant is established. It is the case that all states before the shared events are marked in this example. So the shared event marking condition is satisfied. Practically, it could mean an establishment of a communication system between the decentralized plants in the case that the important events (represented by the shared event) occur. Also, since the all shared events are controllable, the mutual controllability condition is satisfied trivially. Note that again, if the mutual controllability condition is not satisfied in the given system, we could modify the local plants by adding selfloops of the uncontrollable events of other plant to each corresponding local plant. This will not change the structure of the global system, hence as long as the coordination scheme is concerned, this modification will not cause any issues.

In addition, we need to check the local specification E_i is $L_{i,m}$-closed language. From the analysis, it is found to be true. Therefore, all requirements for Preposition 2 in each decentralized plant are now satisfied, i.e., the structural decentralized control is established in this plant.

Now we obtain the coordinator. Firstly, the coordination system $\mathbf{G}_h$ is obtained by natural projection p_h: $\Sigma^* \rightarrow \Sigma_h^*$. Figure 9(a) represents a DES model for the plant $\mathbf{G}_h$. It has been assumed that the system can produce four kinds of

products as presented in Figure 3. The specification E_h for this $\mathbf{G}_h$ is synthesized and displayed in Figure 9(b). Then the supervisor $\mathbf{S}_h$ can be designed for given E_h. The conditions in Theorem 1 can be verified trivially. By Theorem 1, the concurrent actions of $\mathbf{S}_h$ and the local supervisors $\mathbf{S}_i$ will guarantee to achieve the largest possible (optimal) behavior that can be made by the centralized supervisor. Note that the condition given in Theorem 1 is established in the structure of the plant.



(a) **(b)**

Figure 9 DES models for the coordination plant G_h (a) and the Specification E_h (b)

To see the benefits of this structural decentralized coordination scheme, now assume that due to, for example, market demands, the plant is required to produce only two types of products, type I and another type of product, say the product type V. Assume that the product type V uses a combination of two materials B and C with the catalyst E. Also assume that it uses the same procedure as product IV (σ_7) and is drained out through H. Since the specification has changed, in other coordination schemes, we have to verify the whole plant for this new specification since the conditions are specification-dependent. However, in here, the conditions in Theorem 1 is structural, which is established for all $L_{h,m}$-closed language of E_h, (verification for this is much simpler). So the control synthesis can be done easily without any further verification. The modified specification $(E_h)_{\text{new}}$ is shown in Figure 10. A new supervisor $(\mathbf{S}_h)_{\text{new}}$ based on the modified $(E_h)_{\text{new}}$ can be synthesized. By Theorem 1, the concurrent actions of the new $(\mathbf{S}_h)_{\text{new}}$ and $\mathbf{S}_i$ will guarantee to achieve the modified requirements with the global optimality.

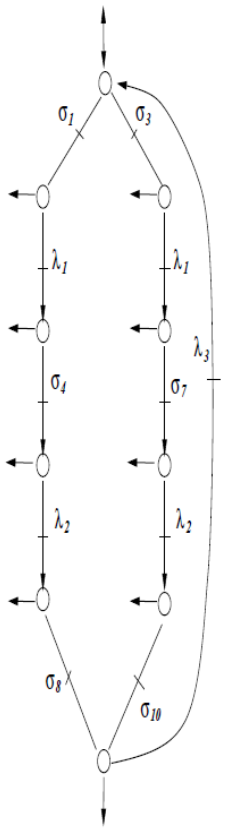


Figure 10 DES models for the modified Specification $(E_h)_{new}$

Conclusion

In this paper, a coordinating scheme in the structural decentralized control framework is presented. The structural conditions to establish the scheme has developed. Under the conditions given in Theorem 1, the combined actions of the decentralized supervisors and the coordinator will achieve the same optimality compared to the centralized control. Also the conditions are applied on the structure of the system. Hence once the verification is made, the coordinating scheme for a set of specifications can be synthesized without any further verification. This definitely can bring a significant computational savings in the case of the plants requiring frequent changes of the specifications, as shown in the example. Extending the results into more general system, for example without the restriction on the control status of the shared events, could be a potential future research area.

Acknowledgement

This work was supported by the University of Incheon (International Cooperative) Research Grant in 2011.

References

[1] Ramadge P. J. & Wonham W. M., "Supervisory control of a class of discrete-event processes", SIAM

J. Control and Optimization, Vol. 25 No. 1, 206–230, 1987

[2] Cassandras C.G. and Lafortune S., *Introduction to Discrete Event Systems*. (2nd. ed.) Kluwer, 2008

[3] Seatzu C., Manuel Silva, and Jan H. van Schuppen (Eds.), "Control of Discrete-Event Systems - Automata and Petri Net Perspectives", Lecture Notes in Control and Information Sciences 433, Springer, 2013

[4] Wonham W. M. & Ramadge P. J., "On the supremal controllable sublanguage of a given language", SIAM J. Control and Optimization, Vol. 25, No. 3, 637-659. 1987

[5] Gohari, P. & Wonham, W. M., "On the complexity of supervisory control design in the RW framework", IEEE transactions on systems, Man, and Cybernetics, Special Issue on DES, Vol. 30, No. 5, 643-652, 2000

[6] Wonham, W. M. & Ramadge, P. J., "Modular supervisory control of discrete event Systems", Mathematics of Control, Signal and Systems, Vol. 1, No. 1,13–30. 1988

[7] Kozak, P. & Wonham, W. M., "Fully decentralized solutions of supervisory control problems", IEEE Transactions on Automatic Control, Vol. 40, No. 12, 2094–2097 1995

[8] Lin, F. & Wonham, W.M., "Decentralized supervisory control of discrete-event systems", Information and Science, Vol. 44, 199-224, 1988

[9] Rudie K & Wonham W. M., "Think globally, act locally: Decentralized supervisory control", IEEE transaction on Automatic control, Vol. 37, No. 11, 1692-1708, 1992

[10] Yoo, T. S. & Lafortune, S., "A general architecture for decentralized supervisory control of discrete-event systems", Discrete Event Dynamic Systems: Theory and Applications, Vol. 12, 335-377, 2002

[11] Lee, S., H. & Wong, K. C. "Structural decentralized control of concurrent discrete-event systems", European Journal of Control, Vol. 8, No. 5, 477-491, 2002

[12] Lin F. & Wonham, W.M., "Decentralized control and coordination of discrete-event systems with partial observation", IEEE transactions on Automatic Control, Vol. 35, No. 12, 1330-1337, 1990

[13] Lin F., "Control of large scale discrete event systems: task allocation and coordination", Systems and Control letters, 17, 169-175, 1991

[14] Komenda J., Masopust T., & van Schuppen J. H., "Synthesis of controllable and normal sublanguages for discrete-event systems using a coordinator", Systems and control letters, Vol. 60, No. 7, 492-502, 2011

[15] Komenda J., Masopust T., & van Schuppen J. H., "Supervisory control synthesis of discrete-event systems using a coordination scheme", Automatica, 48, 247-254, 2012

[16] Akesson, K. & Fabian, M., "Implementation of supervisory control for chemical batch processes", Proceedings of 1999 IEEE International conference on control applications, 1272-1277, 1999

[17] Wonham W. M., *Notes on supervisory control of discrete-event systems*. ECE 1636F/1637S 2013-14, Department of Electrical Engineering, University of Toronto, Toronto, Canada, 2014

[18] Willner Y. & Heymann M., "Supervisory control of concurrent discrete-event systems", International Journal of Control, Vol. 54, No. 5, 1143–1169, 1991

[19] Komenda J., van Schuppen J. H., Gaudin B., & Marchand H., "Supervisory control of modular systems with global specification languages", Automatica, Vol. 44, No. 3, 1127-1134, 2008