International Journal of Precision Engineering and Manufacturing Vol. 5, No. 2, April 2004.

Task Rescheduling Using a Coordinator in a Structural Decentralized Control of Supervisory Control Systems

Sang-Heon Lee [1], Ill-Soo Kim [2,#] and Kai C. Wong [3]

[1] Centre for Advanced Manufacturing Research, University of South Australia, Mawson Lakes, SA, Australia
[2] Dept. of Mech. Eng. Mokpo National University, 61 Dorim-ri, Chungkye-myun, Muan-gun, Chonnam, South Korea
[3] PhD. Dept. of Systems Engineering, RSISE, Australian National University, Canberra ACT 0200, Australia

ABSTRACT

A problem of task rescheduling using a coordinator in a structural decentralized control of supervisory control theory is formulated. we consider that the overall system is divided into a number of local systems. Using an example of a chemical batch reaction process, it has shown that after local supervisors have been established for a given task, a coordinator can be used to solve some rescheduling problems among local plants for new or modified tasks. The coordination system models the interactions of local plants, and is consisting of only the shared events of local plants, so simpler to synthesize. A coordinator is designed based on the specifications given for the coordination system. Under the 'structural' conditions developed in this paper, the combined concurrent actions of the coordinator with the existing local supervisors will achieve the rescheduling requirements. Again since the conditions are structural (not specification-dependent), once the coordination architecture has been established, it can be used for a number of different tasks without further verifications.

Key Words : Discrete-event systems, decentralized control, task rescheduling, Supervisory control system, Chemical batch reactor

1. Introduction

A very large class of dynamic systems, such as the message flow in computer networks or communication networks, the material flow in automated manufacturing systems, and air traffic control systems, comes under the heading of Discrete Event Systems (DES). In contrast to conventional control systems usually described by differential equations, where the changes of underlying state variables are continuously specified over time, DES is characterized by asynchronous, discrete and qualitative changes of state which occur at instants determined by the system dynamics. Many different methods in DES have been developed [1] and each of them has its own problem of interests and limitations. In this paper, we focus on the approach called Supervisory Control Theory (SCT) of DES proposed by Ramadge and Wonham [2,3].

Computational problems in SCT are usually complex. The computational complexity to synthesize supervisors is polynomial in terms of state sizes of the generators representing the plant behaviors and specifications [4]. However, since the system is usually composed of a number of components, the effort to compute a supervisor increases exponentially with the number of components [5,6]. To overcome this problem, within this framework, modular control [4,7], decentralized control [8,9,10], and hierarchical control [11] have been proposed and studied. Most results on decentralized control of SCT are obtained under the assumptions that the overall task has been divided into subtasks and assigned the different local supervisors.

☞ Manuscript received: May 12, 2003 ;
Accepted: November 29, 2003
Corresponding Author:
Email: ilsookim@mokpo.ac.kr
Tel: +82-61-450-2416, Fax: +82-61-452-6376

However, in most works, the conditions, which guarantee that a decentralized control achieves the same behavior as a centralized control would, are specification-dependent. That is, the conditions have to be verified for each given specification. In other words, if a given specification is changed, the conditions should be verified again. However, in a recent research [12], the authors obtained the conditions as 'structural' - not specification-dependent. That is, once those conditions have been verified for a given system structure, then a decentralized control is achieved for a set of specifications (i.e. a number of tasks). That is the reason why this approach is called as a 'structural decentralized control'. It has also shown that this could bring an exponential savings on computational efforts while it still offers the same optimal behavior as that would be obtained by a centralized control.

The concept of coordination in SCT was introduced in [13]. In that paper, the authors consider a situation in which after decentralized supervisors have been designed, a coordinating supervisor, called a coordinator, at a higher or a second level may supervise the interactions of the decentralized supervisors. In the paper [14], the author considers the case that the decentralized control alone cannot achieve the overall task. Using a coordinator (another supervisor), the overall task can be achieved by allocating subtasks to the coordinator and local supervisors.

In this paper, we present a result using these two concepts, coordinator and structural decentralized control. Namely, after a decentralized control design has already been established, a coordinator can be utilized to reschedule certain jobs in local systems to meet different requirements, for example the production of different types of products. The coordination plant model consists of only the shared events among the local plants. For a specification given in the coordination plant, the corresponding supervisor (a coordinator) can be designed. It is shown that under structural conditions similar to the ones in [12], the combined actions of the coordinator with the existing local supervisors will satisfy some rescheduling requirements. Again since the conditions are structural, once the coordination architecture has been established, it can be used for several different tasks. This approach may be useful for a plant which produces several products from a number of materials as will be shown in the example of a chemical batch reactor (Section 3).

2. Problem Statement

In this section we formulate a problem of task rescheduling using a coordinator in a structural decentralized control of DES. For notational simplicity we present the problem formulation in terms of languages.

Firstly we introduce a definition of coordinator. Consider a system as given in Fig.1. Let Σ_i be the event alphabets for the plants Gi for $i = 1,2,\cdots,n$.. Assume that Σ_i is partitioned into controllable and uncontrollable events, i.e. $\Sigma_i = \Sigma_{ic} \cup \Sigma_{iu}$. For two systems, Gi and Gj , where $i, j \in \{1,2,\cdots,n\}$ and i ≠ j, it is allowed that $\Sigma_i \cap \Sigma_j \neq \phi$, and the control status of the shared events is agreed, that is,

$$\Sigma_{iu} \cap \Sigma_j = \Sigma_j \cap \Sigma_{iu} \tag{1}$$

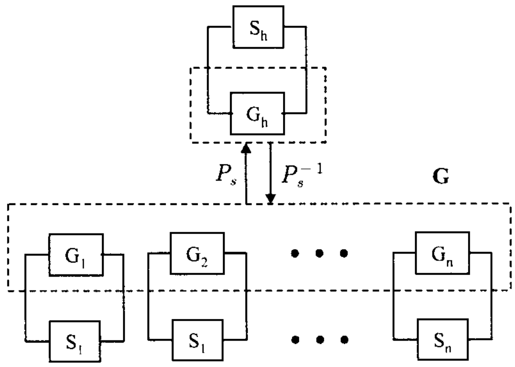

Fig. 1 A coordination scheme

Let $\Sigma := \bigcup_{i=1}^{n} \Sigma_i$ be the event set of the global system, say G. Let $\Sigma_c := \bigcup_{i=1}^{n} \Sigma_{ic}$ and $\Sigma_u := \bigcup_{i=1}^{n} \Sigma_i u$. It can be easily checked that $\Sigma_{iu} = \Sigma_i \cap \Sigma_u$. Let pi be the natural projection from Σ^* to Σ_i^* . Let $L_{i,m}, L_i \subseteq \Sigma_i^*$ represent respectively the marked and closed behaviors of system G_i. We assume that $L_i = \overline{L_{i,m}}$. The marked and closed behaviors of the overall system G are respectively

$$L_m = L_{1,m} \| L_{2,m,} \| \cdots \| L_{n,m} = \bigcap_{i=1}^{n} (p_i)^{-1}(L_{i,m}),$$
$$L = L_1 \| L_2, \| \cdots \| L_n = \bigcap_{i=1}^{n} (pi)^{-1}(Li).$$

Now let $E_i \subseteq L_{i,m}$ be $L_{i,m}$ -closed language, not necessarily prefix-closed, representing a specification on the system Gi. The corresponding specification on G can be taken to be $(pi|_L)^{-1}(E_i)$ (where $(pi|_L)$ denotes the restriction of pi to L), the inverse image of Ei, as done in[8]. The overall specification then is $E := \bigcap_{i=1}^{n} (pi|_L)^{-1}(E_i)$. Local supervisors, denoted Si, can be designed for given local specifications (Ei). The marked and closed behaviors of the closed-loop local system Si/Gi are respectively $_{KL_i}(E_i)$ and $\overline{_{KL_i}(E_i)}$. global supervisor S on the overall specification can be synthesized and $_{KL_i}(E)$ and $\overline{_{KL_i}(E)}$. will be the marked and closed behaviors of the closed-loop system, respectively.

In this paper, we define a coordinator only in the shared event level as follows. Let the shared event set be $\Sigma_s = \bigcup_i \cap (\bigcup_{j \neq i} \Sigma_j)$, and let ps be the natural projection from Σ^* to Σ_s^*. Consider a system G_h such that $L(G_h) = p_s(L)$ and $L_m(G_h) = p_s(L_m)$. For simplicity, we use L_h for $p_s(L)$ and $L_{h;m}$ for $p_s(L_m)$. Let a specification on the system G_h, $E_h \subseteq L_{h;m}$, be an $L_{h;m}$–closed language. Now compute a supervisor S_h for the specification E_h on the plant G_h. Then the marked and closed behaviors of the closed-loop system S_h/G_h are $_{KL_h}(E_h)$ and $\overline{_{KL_h}(E_h)}$, respectively. The modified global specification with E_h is

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

Then we define S_h as a coordinator if

$$\overline{_{KL}(E')} = \bigcap_{i=1}^{n} (p_i|_L)^{-1} \overline{(_{KL_i}(E_i))} \cap (p_s|_L)^{-1} \overline{(_{KL_i}(E_h))} \quad (2)$$

That is, a coordinator is a supervisor on G_h which coordinates the local supervisions such that the combined supervision will achieve the modified global control objective.

What we are interested is the following.

Problem 2.1: For all pairs of the systems $G_1, G_2, \cdots G_n$ and G_h defined as above and for any $L_{h;m}$-closed language E_h, under what condition is it true that the supervisor S_h for E_h on G_h is a coordinator, i.e., Eq. 2 is true?

Thus, the problem is to find conditions under which local syntheses and control of $G_1, G_2, \cdots G_n$ and coordination plant G_h do not result in the loss of optimality compared to the global synthesis.

3. Main Result

To present the main result, we introduce the following definitions.

Firstly, let $H_1, H_2, \cdots H_n$ be languages over the alphabet Σ. Then are $H_1, H_2, \cdots H_n$ nonconflicting if

$$\overline{H_1 \cap H_2 \cap H_3 \cdots \cap H_n} = \overline{H_1} \cap \overline{H_2} \cap \cdots \overline{H_n}.$$

This generalizes the notion of nonconflicting languages [4] to n languages. Then it can be shown that for L as a closed language over Σ and $E_i \subseteq L (i = 1,2,\cdots,n)$, suppose that $\{_{KL}(E_i) | i = 1,2,\cdots,n\}$ are nonconflicting, then

$$_{KL}(\bigcap_{i=1}^{n} E_i) = \bigcap_{i=1}^{n} {_{KL}(E_i)}.$$

Simply this is an extension of Theorem 6.1 in [4].

Secondly, let Σ be an alphabet and H be a language over Σ. Let $\Sigma' \subseteq \Sigma$. We say that H marks Σ' if

$$\Sigma^* \Sigma' \cap \overline{H} \subseteq H \Sigma'$$

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

Finally, let $\Sigma_1, \Sigma_2, \cdots \Sigma_n$ be alphabets, not necessarily pairwise disjoint. Let p_i be the natural projection from Σ^* to Σ_i^* , where $\Sigma = \bigcup_{i=1}^{n} \Sigma_i$. Let Σ_i be partitioned into controllable and uncontrollable events (i.e., $\Sigma_i = \Sigma_{ic} \cup \Sigma_{iu}$) such that Eq. 1 holds. Let $H_i \subseteq \Sigma_i^*$. Then for $i, j \in \{1,2,\cdots,n\}$ and i 6≠ j, we say that H_i and H_j are mutually controllable if

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

where p_i^{ij} and p_j^{ij} are the natural projections from $((\Sigma_i \cup \Sigma_j)^*)$ to Σ_i^* and Σ_j^* comparatively. Two languages H_i and H_j are mutually controllable if H_i is controllable with respect to $p_i^{ij}((p_j^{ij})^{-1}(\overline{H_j}))$ (roughly, this is a model for the 'external' behavior of G_j as seen by G_i) and the shared uncontrollable events $((\Sigma_{ju} \cap \Sigma_i))$, and H_j is controllable with respect to $p_j^{ij}((p_i^{ij})^{-1}(\overline{H_i}))$ and $(\Sigma_{iu} \cap \Sigma_j)$. Then we have the following proposition [16].

Proposition 4.1 Suppose that for $i, j \in \{1,2,\cdots,n\}$ and $i \neq j$,

i) $L_{i,m}$ and $L_{j,m}$ mark $\Sigma_i \cap \Sigma_j$,

ii) Li and L_j are mutually controllable.

Then for any $E_i \in F_{L_{i,m}}$,

$$\bigcap_{i=1}^{n} (p_i|L)^{-1}(\overline{k_{Li}(E_i)}) = \overline{k_L(E)}, \tag{3}$$

where $\overline{k_L(E)}$ is the closed behavior of a global closed-loop system(S/G) and $\overline{k_{Li}(E_i)}$ is the closed behaviors of the closed-loop local system(S_i/G_i).

The proof can be found in [16]. This proposition states that if the marked behaviors of any two subsystems mark their shared events and any two subsystems are mutually controllable, then local syntheses and control for any $L_{i,m}$ -closed specifications do not result in the loss of optimality compared to the global synthesis. This conditions are 'tructural' conditions – not specification-dependent. That is, once those conditions have been verified for a given system structure, then a decentralized control is achieved for a set of specifications (i.e. a number of tasks) without further verification. This practically agrees with the intuition that if the systems are structured properly, then the operation of these systems will be easier. It is possible to show that this could bring an exponential savings on computational efforts compared to a centralized control [12].

Using this proposition, we could have a following theorem which provides sufficient conditions for Problem 3.1.

Theorem 3.1: Let $\Sigma, \Sigma_i, \Sigma_{ic}, \Sigma_{iu}, \Sigma_s, L_{i,m}, L_i (i = 1,2,\cdots,n), L_h, L_{h,m}, L_m$, and L be given as in previous section. Suppose

i) for all pairs of the systems $G_1, G_2, \cdots$, and G_n ($i, j = 1,2,\cdots,n$ and $i \neq j$),

$L_{i,m}$ and $L_{j,m}$ mark $\Sigma_i \cap \Sigma_j$,

ii) for all pairs of the systems $G_1, G_2, \cdots, G_n$ and G_h, L_i and Lj (including L_h)are mutually controllable.

Then Problem 2.1 is solved, that is, any supervisor for a $L_{h,m}$ -closed sublanguage is a coordinator.

If we consider the coordination plant, G_h , as an additional local plant, the conditions in

Theorem 3.1 are very similar to those in Proposition 3.1 except that G_h is not included in the first condition (we call the shared-event-marking condition). This is because some of the structural properties of the local systems are inherited by the coordination plant. That is under the assumptions given in i) of Theorem 3.1, the pair of the systems (G_i, G_h), for $i = 1,2,\cdots,n$, automatically satisfies the shared-event-marking condition as shown in Proposition 3.2.

Proposition 3.2 : Let $\Sigma_i, \Sigma_s, L_{i,m}, L_{h,m}$ be defined as above, for $i = 1,2,\cdots,n$ Then

$L_{i,m}$ and $L_{h,m}$ mark $\Sigma_i \cap \Sigma_s$.

To prove Proposition 3.2, we firstly establish the following lemmas.

Lemma 3.1: Assume that $L_i, L_{i,m}$ and p_s are given as above where. Then $i = 1,2,\cdots,n$

$$p_s(L_i) = p_s(L_{i,m}).$$

Proof: Since one inclusion is always true, one only needs to show that $p_s(L_i) \subseteq p_s(L_{i,m})$.

Let $s \in p_s(L_i)$. Then there exists a string $u \in L_i$ such that $s = p_s(u)$. Since $L_i = \overline{L_{i,m}}$,

there exists a string $v \in \Sigma_i^*$ such that $uv \in L_{i,m}$ Hence $p_s(uv) \in p_s(L_{i,m})$. We consider two case. If $v \in (\Sigma_i - \Sigma_s)^*$, then $p_s(uv) = p_s(u)p_s(v) = p_s(u) = s \in p_s(L_{i,m})$. Otherwise, there are events $\sigma_1, \sigma_2, \cdots \sigma_n \in \Sigma_s$ and $\omega_1, \omega_2, \cdots, \omega_n, \omega_{n+1} \in (\Sigma_i - \Sigma_s)^*$ such that $v = \omega_1\sigma_1\varpi_2\sigma_2 \cdots \omega_n\sigma_n\omega_{N+1}$, Since $, uv \in L_{i,m}, uv = u\omega_1\sigma_1\varpi_2\sigma_2 \cdots \omega_n\sigma_n\omega_{n+1} \in L_{i,m}$. Hence $u\omega_1\sigma_1 \in \overline{L_{i,m}}$. Also, since $u \in L_i$ and $\omega_1 \in (\Sigma_i - \Sigma_s)^*, u\omega_1 \in \Sigma_i^*$. Since $\sigma \in \Sigma_s = \bigcup_{i \neq j}(\Sigma_i \cap \Sigma_j), \sigma \in \Sigma_i \cap \Sigma_j$ for some j. By the assumption of the shared event marking condition between two systems G_i and G_j, one has that $u\omega_1 \in L_{L_{i,m}}$. Therefore,

$$p_s(u\omega_1) = p_s(u)p_s(\omega_1) = p_s(u) = s \in p_s(L_{i,m})$$

Next two lemmas are used to establish Lemma 3.4

Lemma 3.2 : [15] For alphabets $\Sigma_0, \Sigma_1, \Sigma_2$ with $\Sigma = \Sigma_1 \cup \Sigma_2$ and $\Sigma_0 \subseteq \Sigma$, let $L_1 \subseteq \Sigma_1^*, L_2 \subseteq \Sigma_2^*$ and let $P_0 : \Sigma^* \to \Sigma_0^*$ be the natural projection. Then $P_0(L_1 \| L_2) \subseteq (P_0 L_1) \| (P_0 L_2)$. Note that $P_0 L_i \subseteq (\Sigma_0 \cap \Sigma_i)^*$.

Proof: Recall that $L_1 \| L_2 = p_1^{-1}(L_1) \cap p_2^{-1}(L_2)$, where $p_i : \Sigma^* \to \Sigma_i^*$ and

$(P_0 L_1) \| (P_0 L_2) = (p_1^0)^{-1}(p_0(L_1)) \cap (p_2^0)^{-1}(p_0(L_2))$, where $p_i^0 : \Sigma_0^* \to (\Sigma \cap \Sigma_i)^*$.

To show $P_0(L_1 \| L_2) \subseteq (P_0 L_1) \| (P_0 L_2)$, consider a string $s \in P_0(L_1 \| L_2)$. Then there exists a string $u \in L_1 \| L_2$ such that $s = p_0(u)$. Hence $u \in p_1^{-1}(L_1) \cap p_2^{-1}(L_2)$ and $s = p_0(u)$. So, $p_1(u) \in L_1$ and $p_2(u) \in L_2$. Therefore, $p_0 p_1(u) \in p_0(L_1)$ and $p_0 p_2(u) \in p_0(L_2)$. Note that in this case $p_i^0(s) = p_i^0 p_0(u) = p_i p_0(u) = p_0 p_i(u)$. Hence, $p_1^0(s) \in p_0(L_1)$ and $p_2^0(s) \in p_0(L_2)$. So,

$$s \in (p_1^0)^{-1} p_0(L_1) \cap (p_2^0)^{-1} p_0(L_2) = (P_0 L_1) \| (P_0 L_2).$$

Lemma 3.3 : Let $\Sigma_0, \Sigma_1, \Sigma_2, p_0, p_i, p_i^0$ be defined as in Lemma 3.2. Suppose that

$\Sigma_1 \cap \Sigma_2 \subseteq \Sigma_0$.

Then

$p_0(L_1 \| L_2) = (P_0 L_1) \| (P_0 L_2)$.

Proof: One inclusion ($\subseteq$) is already shown in Lemma 3.2.

Lemma 3.4 : Let Σ_i be the event alphabets for the plant G_i for $i = 1, 2, \cdots, n$ and let $\Sigma := \bigcup_{i=1}^{n} \Sigma_i$. It is allowed that $\Sigma_i \cap \Sigma_j \neq \phi$ for $i, j \in \{1, 2, \cdots, n\}$ and $i \neq j$. Let $\Sigma_s = \bigcup_{i \neq j} (\Sigma_i \cap \Sigma_j)$, and let p_i and p_s be the natural projection from Σ^* to Σ_i^* and to Σ_s^*, respectively.

Then for $L_i \subseteq \Sigma_i^*$,

$$p_s(L_1 \| L_2 \| \cdots \| L_n) = p_s(L_1) \| p_s(L_2) \| \cdots \| p_s(L_n).$$

Proof: One has that

$p_s(L_1 \| L_2 \| \cdots \| L_n)$

$= p_s(L_1) \| p_s(L_2 \| \cdots \| L_n)$

(since $\Sigma_1 \cap (\bigcup_{j \neq 1} \Sigma_j) \subseteq \Sigma_s$ and by Lemma 3.3)

$= p_s(L_1) \| p_s(L_2) \| p_s(L_3 \| \cdots \| L_n)$

(by Lemma 3.3)

$= p_s(L_1) \| p_s(L_2) \| \cdots \| p_s(L_{n-1} \| L_n)$

(by Lemma 3.3)

$= p_s(L_1) \| p_s(L_2) \| \cdots \| p_s(L_n)$.

It follows from Lemmas 3.1 and 3.4, we have the following.

Lemma 3.5 : Let $L_{h,m} = P_s(L_m)$ and $L_h = p_s(L)$. Then $L_h = L_{h,m}$.

Proof: It can be shown in turn

$L_h = p_s(L_1 \| L_2 \| \cdots \| L_n)$

$= p_s(L_1) \| p_s(L_2) \| \cdots \| p_s(L_{n,m})$ (by Lemma 3.4)

$= p_s(L_{1,m} \| p_s(L_{2,m}) \| \cdots \| P_s(L_{n,m})$ (by Lemma 3.1)

$= p_s(L_{1,m} \| L_{2,m} \| \cdots \| L_{n,m})$ (by Lemma 3.4)

$= L_{h,m}$

Now we prove Proposition 3.2.

Proof of Proposition 3.2: It is obvious that $L_{i;m}$ marks $\Sigma_i \cap \Sigma_s$ since we assume that $L_{i;m}$

For $i = 1, 2, \cdots, n,$, marks its shared events. Given Lemma 3.5, $L_{h;m}$ also marks $\Sigma_i \cap \Sigma_s$.

The proposition 3.2 shows that some of the structural properties of the given subsystems Gi are inherited by the coordination plant G_h. Now, theorem 3.1 can be proved by Proposition 3.1 and 3.2.

4. Example : Chemical batch reactor

In this section, we consider an example to illustrate the results. All the computations in this section are carried out with the software package CTCT developed along with the notes by W. M. Wonham [15]. Consider a chemical batch reactor shown in Fig. 2. The reactor is comprised of the following elementary components: the reaction tank, three material feed valves V_1, V_2 and V_3, three material feed pumps P_1, P_2 and P_3, three drain valves V_4, V_5 and V_6, three drain pumps P_4, P_5 and P_6, two supply valves for catalysts V_7 and V_8, a low and high level sensors WL_1 and WH_1 for the tank, a heater with a temperature probe TP_1, a continuous reaction controller C_1 and two timers T_1 and T_2. 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. It is assumed that the reactor can produce several different products depending on different materials and catalysts supplied as well as the process methods chosen.

The batch process operates as follows. Firstly, the necessary feed valves are open depending on the required materials (A, B or C) and then the corresponding feed pumps are turned on to fill the reactor tank with materials until WH_1 indicates that the volume is 100l. Then the heater is turned on to warm up the material. The temperature and the duration of the heating operation depend on the production methods. We assume that there are four different production methods (say Method I to IV). Also in this stage, either catalyst D or E will be added to the tank for the faster reaction. When the reaction is completed, one of the drain valves (F, G or H) are open and the corresponding drain pump is turned on to discharge the finished product from the tank until WL_1 indicates that the tank is empty. When this happens, the drain pump is turned off and the drain valve is closed. This completes one cycle of the batch process.

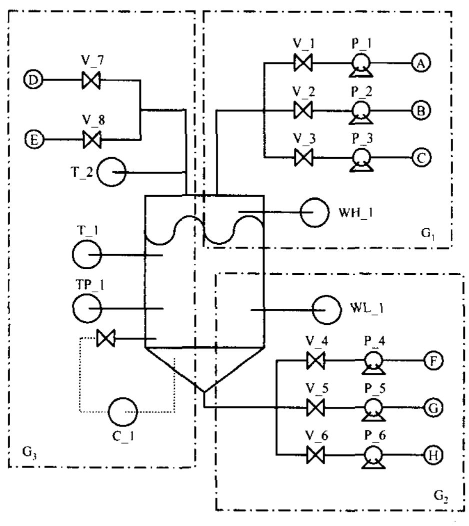

Fig. 2 Batch reactor diagram

For a decentralized control, the whole plant can be naturally decomposed into three subplants as shown in Fig. 2: the filling subplant (G_1), the draining subplant (G_2) and the reaction subplant (G_3). Note that there are no shared components among subplants. Also, the whole operation can be divided naturally into three subprocesses according to the decomposition of the plant: the filling process, the draining process and the reaction process. DES models of elementary components of each plant are shown in Figs. 3, 4 and 5, respectively. Here a state is represented by a circle and an event is described by an arrow from an exit state to an entrance state with an event label attached. The initial state is labeled with an entering

arrow ($\rightarrow\circ$), and a marker state is labeled with an exiting arrow ($\rightarrow\circ$) A double arrow ($\leftrightarrow\circ$) indicates that the initial state is also a marker state. The arrow with a 'tick' indicates that the event is controllable.

It is further assumed that a combination of any two materials is used for obtaining one kind of product. We also assume that the system produces four different kinds of products as follows. The product type I is produced from the mixed materials A and B with the catalyst D and by the reaction method I, and drained into the exit F. The product type II is from the materials A and C with the catalyst E and by the reaction method II, and drained into the exit G. The product type III is from the materials B and C with the catalyst D and by the reaction method III, and drained into the exit F while the type IV is from the materials A and B with the catalyst E and by the reaction method IV, and drained into the exit H.

To enforce that the three subprocesses are operating serially, we introduce controllable shared events which indicate the beginning or the ending of a certain process. For example, the shared event σ_1 is to indicate that only the materials A and B are used for the production, while σ_2 and σ_3 indicate that the materials B and C, and C and A are used, respectively. The events $\sigma_4, \sigma_5, \sigma_6, \sigma_7$ are representing the production methods I to IV used, respectively and $\sigma_8, \sigma_9, \sigma_{10}$, are to indicate which drain exit (F,G or H) is used. Also λ_1, λ_2, and λ_3 are introduced to indicate a completion of the filling process, the reaction process and the draining process (i.e. one cycle of the whole process), respectively. These controllable shared synchronized events are represented as DES models, called 'flag's, and considered as a part of the plants. The DES model for flag_1 given in Fig. 6(b) represents that after the shared events (σ_1, σ_2, or σ_3) have occurred, 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.

There also exist some physical constraints among the elementary components and they will restrict the system behavior by deleting infeasible states and transitions. Physical constraints are often resulted from conservation of mass, energy and momentum, gravitational consideration and mechanical connections among the components. For example, for the case of plant G_1, the level in the tank cannot be increased after all the feed valves have been closed. The DES model for this is given in Fig. 6(a). Those for plants G_2 and G_3 can be obtained in a similar way. The DES model for each plant is obtained by synchronous composition of the elementary components, the flag and the physical constraint.

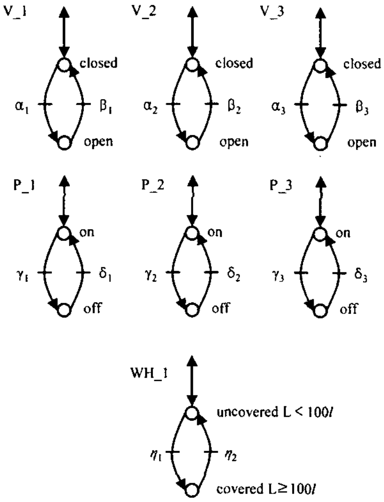

Fig. 3 DES models for elementary components of G_1

Now we establish a DES model for the specifications for each plant. Since it is assumed that the mixture of any two materials is used for the chemical reaction, the specification for the plant G_1, denoted by E_1, is as follows:

1. For the mixture of the materials A and B, after the event σ_1 (indicating this mixture) has occurred, 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). Note that the order is important to prevent the motor overloads.

2. For the mixture of the materials B and C, after the event σ_2 has occurred, 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, after the event σ_3 has occurred, open the valves V_1 and V_3 (α_3, α_4) and turn on the pumps P_1 and P_3 (γ_1, γ_3).

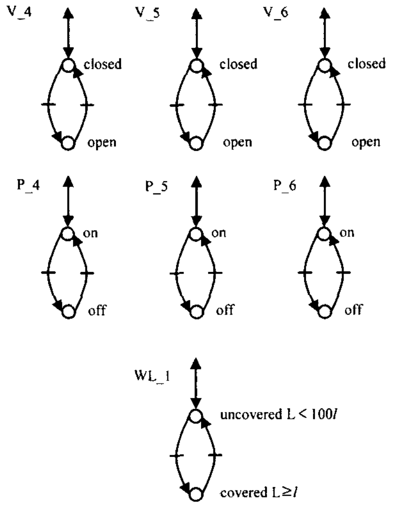

Fig. 4 DES models for elementary components of G_2

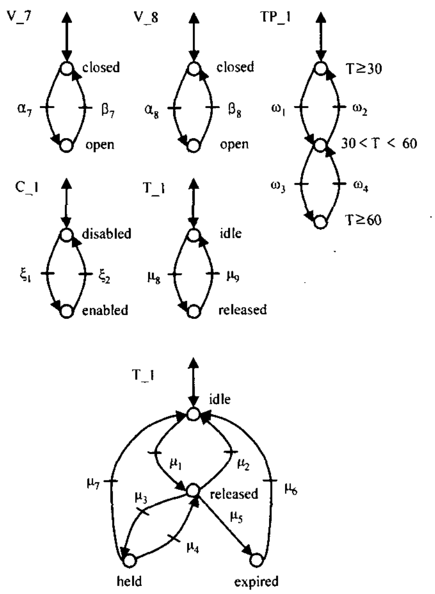

Fig. 5 DES models for elementary components of G_3

4. When the level in the tank reaches $L \geq 100l(\eta_1)$ turn off the pumps ($\delta_1, \delta_2, \delta_3$) and close the valves ($\beta_1, \beta_2, \beta_3$) whichever necessary. Then send a

signal (λ_1) to the other plants.

5. The subplant G1 should not restrict the behaviors of the other subplants.

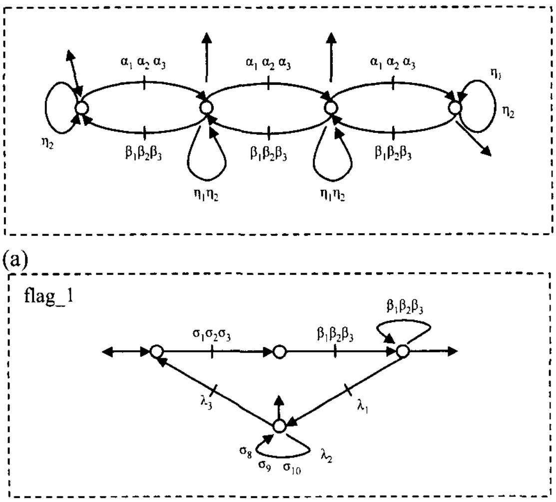

Fig. 6 DES models for physical constraint and synchronization flag for the plant G_1

1. After the shared event σ_4 has occurred, increase the temperature of mixed material in the tank to $30°C\langle T\langle 60°(\omega_1)$. Then enable the reaction controller C_1 (ξ_1).

2. Release the timer T_1. Assume that the controller is required to operate for 10 minutes at the material temperature $30°C\langle T\langle 60°$. So μ_1 is chosen.

3. During the operation, the timer can be held at its current time (μ_3) to handle some situations such as emergency repairs. From the held state, the timer can either be reset (μ_7) to the idle state or re-released (μ_4).

4. After the set time has expired (uncontrollable event μ_5), the controller C_1 is disabled (ξ_2) and the timer T_1 is reset (μ_6).

5. To supply the catalyst D, open the valve V_7 (α_7); then release the 30-second timer T 2 (μ_8).

6. After T_2 has expired (uncontrollable event μ_9), close the valve V_7 (β_7).

7.Then increase the temperature to $T\rangle 60°C(\omega_3)$ and enable the reaction controller C_1 (ξ_1) again.

8. Release the timer T_1. At this time, we assume that the reaction is required to operate for 15 minutes. So μ_2 is chosen.

9. When the time has expired (μ_5), the controller C_1 is disabled (ξ_2) and the timer is reset (μ_6).

10. Wait until the material temperature is cooled down to $T\langle 30°C(\omega_4,\omega_2)$. .

11. The subplant G_3 should not restrict the behaviors of the other subplants.

The requirements for the other methods, II, III and IV, can also be established similarly. DES models for this is given in Fig. 8. Now the local supervisors, S_1 for E_1, S_2 for E_2, and S_3 for E_3 are then computed.

The shared event set is and consisted of only controllable events. Now we obtain the coordination system G_h by natural projection $p_s:\Sigma^*\rightarrow\Sigma_s^*$.

$$\Sigma_s=\{\sigma_1,\sigma_2,\sigma_3,\sigma_4,\sigma_5,\sigma_6,\sigma_7,\sigma_8\sigma_9\sigma_{10},\lambda_1,\lambda_2,\lambda_3\},$$

Fig. 9(a) represents a DES model for the plant G_h. It has been assumed that the system can produce four kinds of products as described earlier. According to this requirement, the specification for G_h, E_h, can be established and a DES model for this is displayed in Fig. 9(b). Then the supervisor S_h can be designed for given E_h.

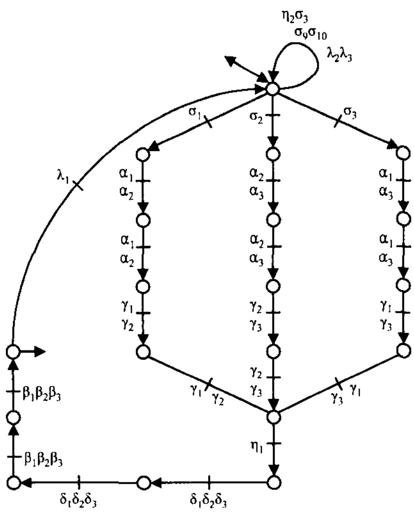

Fig. 7 DES models for the specification E_1 for the subplant G_1

We now verify that the system structure of this example (G_1, G_2, G_3 and G_h) satisfies the required conditions in Theorem 3.1. Since all the shared events are controllable, the mutual controllability conditions are satisfied trivially. For the 'shared event marking' condition, we mark all the states before the shared events. It is the case that the local specification E_i is $L_{i;m}$–closed language. Hence the conditions are satisfied in this example. Therefore, by Theorem 3.1, the concurrent actions of S_h and S_i will guarantee to achieve the specification.

Now to show the benefits of the structural decentralized control, assume that due to, for example, market demands, the system 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 is made from the mixed materials B and C with the catalyst E by the reaction method IV and then drained through the drain exit H.

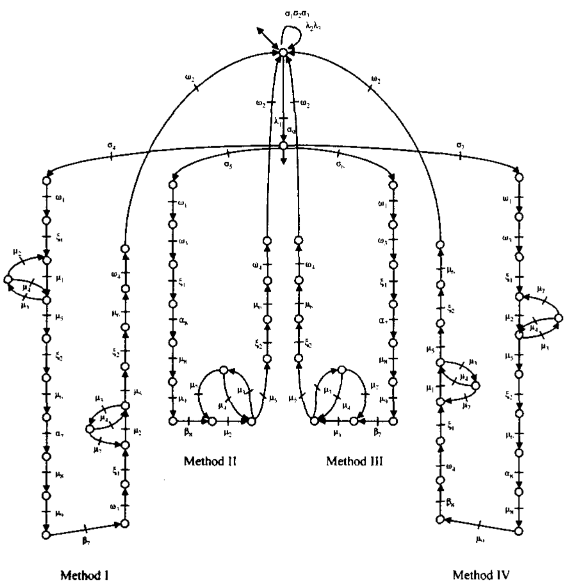

Fig. 8 Specification E_2 for the plant G_2

To meet with new requirements, without structural decentralized control, the local supervisors for each local plant need to be redesigned. But with structural decentralized control, it can be done in the plant Gh by simply allowing or prohibiting certain shared events from occurring. The specification Eh is changed accordingly and shown in Fig. 9(c). A new supervisor Sh based on the modified is Eh designed. By Theorem 3.1, the concurrent actions of the new Sh and Si will guarantee to achieve the modified requirements.

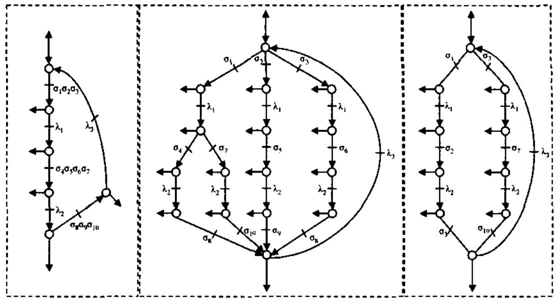

Fig. 9 A coordination plant G_h

5. Conclusions

In this paper, we have shown that after the local supervisors have been designed for a given task, a coordinator can be used to solve some rescheduling problems among local plants G_i. The coordination system G_h models the interactions of local plants, and is usually simpler than the local plants. Then a coordinator is designed based on the specification given for the system G_h. Under the conditions given in Theorem 3.1, the combined concurrent actions of the coordinator with the existing local supervisors will achieve some rescheduling requirements.

By using different coordinators, we have shown that under the conditions, it is possible to achieve the control objectives for different tasks without further verification, as seen in the example provided. As a future work, it might be useful to extend the results into more general system, for example without the restriction on the control status of the shared events.

References

1. Cassandras, C. and Lafortune, S., Introduction to Discrete Event Systems. Kluwer Academic Pub., 1999.
2. Ramadge, P. J. and Wonham, W. M., "Supervisory Control of a Class of Discrete-Event Processes," SIAM J. Control and Optimization, Vol. 25, No. 1, pp. 206–230, 1987.
3. Wonham, W. M. and Ramadge, P. J., "On the Supremal Controllable Sublanguage of a Given Language," SIAM J. Control and Optimization, Vol. 25, No. 3, pp. 637–659, 1987.

4. Wonham, W. M. and Ramadge, P. J., "Modular Supervisory Control of Discrete Event Systems," Mathematics of Control, Signal and Systems, Vol. 1, No. 1, pp. 13–30, 1988.
5. Ramadge, P. J., "The Complexity of Some Basic Control Problems for Discrete-Event Systems," Advanced computing concepts and techniques in control engineering, pp. 171–190, 1988.
6. Ramadge, P. J. and Wonham, W. M., "The Control of Discrete Event Systems," Proc. IEEE, Special Issue on Discrete Event Dynamic Systems, Vol. 77, No. 1, pp. 81–98, 1989.
7. Ramadge, P. J. and Wonham, W. M., "Modular Feedback Logic for Discrete Event Systems," SIAM J. Control and Optimization, Vol. 25, No. 5, pp. 1202–1218, 1987.
8. Lin, F. and Wonham, W. M., "Decentralised Supervisory Control of Discrete-Event Systems," Information Sciences, Vol. 44, pp. 199–224, 1988.
9. Willner, Y. and Heymann, M., "Supervisory Control of Concurrent Discrete-Event Systems," International Journal of Control, Vol. 54, No. 5, pp. 1143–1169, 1991.
10. Rudie, K. and Wonham, W. M., "Think Globally, Act locally: Decentralized Supervisory Control," IEEE Transactions on Automatic Control, Vol. 37, pp. 1692–1708, November 1992.
11. Zhong, H. and Wonham, W. M., "On the Consistency of Hierarchical Supervision in Discrete-Event Systems," IEEE Transactions on Automatic Control, Vol. 35, pp. 1125–1134, October 1990.
12. Lee, S. H. and Wong, K. C., "Decentralised Control of Concurrent Discrete-Event Systems with Non-Prefix Closed Local Specifications," in Proceedings of the 36th IEEE Conference on Decision and Control, pp. 2958–2963, December 1997.
13. Lin, F. and Wonham, W. M., "Decentralized Control and Coordination of Discrete-Event Systems with Partial Observation," IEEE Transactions on Automatic Control, Vol. 35, No. 12, pp. 1330–1337, 1990.
14. Lin, F., "Control of Large Scale Discrete Event Systems: Task Allocation and Coordination", Systems and Control Letters, Vol.17, pp. 169–175, 1991.
15. Wonham, W. M., "Notes on Control of Discrete-Event Systems," Department of Electrical Engineering, University of Toronto, 1999.
16. Lee, S. H., "Decentralised Control of Concurrent Discrete-Event Systems," Canberra, Australia: PhD thesis, The Australian National University, 1998.
17. Chung, B. M., Lee, J. W., Joo, H. H. and Lim, Y. K., "Hybrid Fuzzy Learning Controller for an Unstable Nonlinear System." International Journal of the Korea Society of Precision Engineering, Vol. 1, No. 1, pp.79-83, 2000.