DEVELOPMENT OF CONCURRENT STRUCTURAL
DECENTRALISED DISCRETE EVENT
SYSTEM USING BISIMULATION CONCEPT

SANG-HEON LEE
The School of Engineering
The University of South Australia, Mawson Lakes, SA 5095 Australia

ABSTRACT. This paper presents a concurrent structural decentralised control in the framework of supervisory control theory using bisimulation concept. It is a way to weaken the shared-event-marking condition of structural decentralised control developed by Lee and Wong [7]. The sufficient conditions to guarantee the global optimality achieved by the concurrent actions of simpler decentralised control have been presented. The developed condition becomes specification dependent, however, the other structural condition, the mutual controllability condition, is still applied on the structure of the system. Hence the computational savings are still achievable. An example is provided to illustrate the result.

1. Introduction. Discrete-event system is characterised by discrete and qualitative changes of state values with abrupt occurrences of events over time rather than by ticks of a clock. Since their nature is quite different from conventional time-driven systems, a formal method to analyse and design such systems is necessary. One of such methods is called Supervisory Control Theory (SCT) proposed by Ramadge and Wonham [10] and Wonham [15]. For the last three decades, many follow-up researches within this framework have been conducted. However the applications of SCT into real, practical industrial applications are much smaller than the academic research. One of the major obstacles of such reality is in its computational complexity. Even though the computational complexity to synthesise supervisors in SCT is only polynomial in terms of state sizes of the generators representing the plant behaviours and specifications, the effort to compute a supervisor increases exponentially with the number of components [3]. To overcome this problem, within this framework, modular or decentralised control [2, 5, 8, 11, 17] and hierarchical control [4, 12, 16] have been proposed and studied. However the computational complexity still remains as a critical issue since the whole plants need to be checked for the eligibility conditions of the scheme for every single specification given. A structural decentralised approach in SCT [7] has proposed to overcome some of this issue. The conditions developed in their research are applied to the structure of the system, i.e., the conditions to guarantee the concurrent operations of the decentralised plants to be the same optimal as the centralised control operation are validated in the structure of the system. Hence once the structural decentralised control is established, the future minor changes of specifications (like change of sequences of the operations) can be made directly without any computationally expensive validation process. This obviously brings a great amount of computational

2010 Mathematics Subject Classification. Primary: 93C55; Secondary: 93C83.
Key words and phrases. Discrete event system; supervisory control theory; decentralised control; structural conditions; bisimulation.
savings in the long term operations. Basically, it coincides with our intuitive thinking, i.e., if the structure of the system is properly arranged with more flexibility, then subsequent operations would be much easier. Two sufficient conditions on the structure of the system are developed: all shared events concurrently operated in more than two local plants should be marked (called the shared-event-marking condition) and each decentralised plant should be able to track down the occurrences of any uncontrollable shared events happening in other plants (called the mutual controllability condition). These two conditions are intuitively correct since the first condition implicates that each decentralised plant should be ready before any cooperation with other decentralised plants, and the second condition indicates that any unavoidable events like machine breakdown or electricity black-out happened in other decentralised plants should be possible to be monitored from other plants so that the corresponding action in each decentralised plant can be taken. It has been shown in a more detailed analysis that by achieving these two structural conditions, there is generally an exponential saving of the computational efforts involved [7].

However from the further investigations, it has found that the shared-event-marking condition is too strong, i.e., some states are needed to be marked unnecessarily. As one of natural ways to extend the research, we are seeking to find a way to weaken this condition. It is interesting to note that the finite state machine represented by automata in SCT contains the sequential information as a language and the branching structure of allowable events. Fundamentally, due to the detailed level of information provided by the framework of SCT, it is possible to allow SCT problems to be generalised to several possible process-theoretical semantics. One of potential semantics that can be and has been used in DES is the bisimulation semantic. This motivates us to explore the concept of bisimulation to weaken the shared-event-marking condition. In fact, bisimulation has originally developed to deal with the behavioural equivalency within state transition systems [9] and been used successfully in many applications including discrete-event systems. The research conducted in [1] discussed the relationship between bisimulation and controllability of SCT and showed that bisimulation concept can be used to solve basic SCT problems especially in input/output finite state machine model matching. Zhou and others [18] proposed a necessary and sufficient condition for the existence of a bisimilar supervisor in the SCT framework and the further investigation [19] has developed a way to construct a deterministic bisimilar supervisor in this framework. In the research of Sun etc. [13], a new framework to address the decentralised bisimilarity SCT has developed and a synthesis method to establish decentralised bisimilarity supervisors is introduced.

With the similar point of view, this paper adopts the concept of bisimulation to solve decentralised DES, but with a different aspect: we adopt it to relax one of conditions in structural decentralised control. Since the benefit of structural decentralised control is obvious, it would be desirable to keep its basic framework as much as possible while bisimulation would help to reduce unnecessary actions to establish structural decentralised DES. As a result, it has found the decentralised control condition using bisimulation now becomes specification dependent like other works. However the second condition (mutual controllability condition) is still applied into the structure. So whenever a specification is modified, only the condition using bisimulation concept needs to be verified, which is simpler to be done.

The paper is organised as follows. Section 2 introduces some basic concepts of SCT and bisimulation and Section 3 formulates the problem for the paper. Section 4
presents a concurrent decentralised control of discrete-event systems in supervisory control theory using bisimulation concept. An example is presented for illustration in Section 5 and the paper ends with some concluding remarks in Section 6.

2. Preliminary. In the framework of supervisory control theory, a 5-tuple automaton is used to model a finite state DES. Basically, it is a generator modelled as a set of finite sequence of events representing the behaviour of DES: \( G = (Q, \Sigma, \delta, q_0, Q_m) \), where \( Q \) is a set of finite states, \( \Sigma \) is a finite set of event labels (called an alphabet), \( \delta: Q \times \Sigma \rightarrow Q \) 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. It is common to define the empty string \( \epsilon \notin \Sigma \) to represent the alphabet with no events. Any set of finite strings of event labels in \( \Sigma \) and \( \epsilon \) is defined as \( \Sigma^* \), a language over \( \Sigma \). Any feasible language generated from \( G \) is called the closed behaviour, \( L(G) \), and a special subset of such languages which reaches to the marker states is called the marked behaviour, \( L_m(G) \). Their formal definitions are, respectively: \( L(G) := \{ s \in \Sigma^* | \delta(q_0, s) \text{ is defined} \} \), and \( L_m(G) := \{ s \in L(G) | \delta(q_0, s) \in Q_m \} \). Obviously, \( L_m(G) \subseteq L(G) \). For two strings \( s, u \in \Sigma^* \), \( s \) is a prefix of \( u \) if there is a string \( v \in \Sigma^* \) such that \( u = sv \). The prefix closure of \( K \subseteq \Sigma^* \) is defined by the set of all prefixes of strings in \( K \), i.e., \( K = \{ s \in \Sigma^* | \exists \nu \in \Sigma^* \text{ such that } sv \in K \} \). The language \( K \) is closed if \( \overline{K} = K \). Clearly, \( L(G) \) is closed but \( L_m(G) \) may not. Let \( F \subseteq \Sigma^* \). Then, the language \( F \) is said to be \( H \)-closed if \( F = \overline{F} \cap H \). Then a language \( F \) is nonblocking with respect to \( H \) if \( \overline{F} = \overline{F} \cap \overline{H} \). Certainly if \( G \) satisfies \( L(G) = \overline{L_m(G)} \), then \( L(G) \) is non-blocking. For two languages, \( H_1, H_2 \subseteq \Sigma^* \), if \( \overline{H_1 \cap H_2} = \overline{H_1} \cap \overline{H_2} \), they are said to be nonconflicting.

Let \( G_i := (Q_i, \Sigma_i, \delta_i, q_{i0}, Q_{1,m}) \), for \( i = 1, 2 \). Then the product of two automata \( G_1 \) and \( G_2 \) is given by \( G_1 \times G_2 = Rch(Q_1 \times Q_2, \Sigma_1 \cup \Sigma_2, \delta_1 \times \delta_2, (q_{10}, q_{20}), Q_{1,m} \times Q_{2,m} ) \) according to

\[
\delta_1 \times \delta_2((q_1, q_2), \sigma) = (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma)) \quad \text{if } \delta_1(q_1, \sigma) \text{ is defined and } \delta_2(q_2, \sigma) \text{ is defined},
\]

Undefined otherwise.

Note that \( Rch(\cdots) \) means the reachable component of a DES. Intuitively, the product allows only common event labels to occur simultaneously. Also it is clear that \( L(G_1 \times G_2) = L(G_1) \cap L(G_2) \) and \( L_m(G_1 \times G_2) = L_m(G_1) \cap L_m(G_2) \).

The synchronous composition of two automata is defined by \( G_1 || G_2 = Rch(Q_1 \times Q_2, \Sigma_1 \cup \Sigma_2, \delta_{12}, (q_{10}, q_{20}), Q_{1,m} \times Q_{2,m} ) \), according to, for \( \sigma \in \Sigma_1 \cup \Sigma_2 \),

\[
\delta_{12}((q_1, q_2), \sigma) = (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma)) \quad \text{if } \delta_1(q_1, \sigma) \text{ is defined and } \delta_2(q_2, \sigma) \text{ is defined},
\]

\[
(\delta_1(q_1, \sigma), q_{2}) \quad \text{if } \delta_1(q_1, \sigma) \text{ is defined and } \delta_2(q_2, \sigma) \text{ is not defined},
\]

\[
(q_1, \delta_2(q_2, \sigma)) \quad \text{if } \delta_1(q_1, \sigma) \text{ is not defined and } \delta_2(q_2, \sigma) \text{ is defined},
\]

Undefined otherwise.

The synchronous composition of two automata synchronises the shared events while unshared events can occur whenever possible.

The natural projection is defined to remove events from a string which do not belong to a particular event set as follows: for \( \Sigma_1 \) and \( \Sigma_2 \), where \( \Sigma_1 \cap \Sigma_2 \neq \phi \) and
\( \Sigma = \Sigma_1 \cup \Sigma_2 \), the natural projection \( p_i : \Sigma^* \to \Sigma_i^* \) is defined by, for \( s \in \Sigma^* \) and \( \sigma \in \Sigma_i \),

\[
p_i(\epsilon) = \epsilon, \text{ and } p_i(s\sigma) = p_i(s)\sigma \text{ if } \sigma \in \Sigma_i,
p_i(s) \text{ Otherwise.}
\]

Using this, the closed and marked behaviours of synchronous composition can be obtained: \( L(G_1||G_2) = p_1^{-1}(L(G_1)) \cap p_2^{-1}(L(G_2)) \) and \( L_m(G_1||G_2) = p_1^{-1}(L_m(G_1)) \cap p_2^{-1}(L_m(G_2)) \), where \( p_i^{-1} \) is the inverse projection of \( p_i \).

The control of DES is introduced as follows. We divide the whole event set into two: the controllable events \( \Sigma_c \subseteq \Sigma \), and the remaining uncontrollable events \( \Sigma_u := \Sigma - \Sigma_c \). The controllable events are those which can be disabled or enabled by supervisors while the uncontrollable events are those cannot, i.e., always enabled.

The desired behaviour of \( G \) according to the specification \( E \) can be generated by enabling or disabling controllable events whenever necessary. Wonham and Ramage [10] developed an algorithm to obtain a supervisor \( S \) which can generate a maximal permissive such event: \( L_m(S/G) = \kappa L(G)(L_m(E) \cap L_m(G)) \). The computational effort to obtain \( S \) is \( O(n^2m^2) \), where \( n \) is the number of states in \( G \), and \( m \) is the number of states in \( E \) [10].

For two transition systems, the concept of bisimulation is one way to describe an equivalence relation between them. Let \( \Sigma_i \) be the event alphabet of a subplant \( G_i \), for \( i = 1,2 \). It is allowed that \( \Sigma_i \cap \Sigma_j \neq \emptyset \) for \( i \neq j \). Let \( \Sigma := \Sigma_1 \cup \Sigma_2 \) and \( \Sigma_s := \Sigma_1 \cap \Sigma_2 \). Let \( p_i \) be the natural projections from \( \Sigma^* \) to \( \Sigma_i^* \). For two languages, \( H_i \subseteq \Sigma_i^* \) for \( i = 1, 2 \), we define a binary relation \( \equiv_{H_1,H_2} \subseteq \overline{\Pi_1} \times \overline{\Pi_2} \) as follows [9]:

**Definition 1.** For \( s_1 \in \overline{\Pi_1} \) and \( s_2 \in \overline{\Pi_2} \), we say that \( s_1 \equiv_{H_1,H_2} s_2 \), if and only if \( p_i(s_1) = p_i(s_2) \).

Now we recall a concept of bisimulation [9].

**Definition 2.** Let \( S \subseteq \overline{\Pi_1} \times \overline{\Pi_2} \). We say that \( S \) is a bisimulation on \( H_1 \) and \( H_2 \) with respect to \( \Sigma_1 \cap \Sigma_2 \), if for \( (s_1, s_2) \in S \),

\[
i) (\forall \sigma_1 \in \Sigma_1), s_1\sigma_1 \in \overline{\Pi_1} \text{ implies } (\exists u \in \Sigma_2^*), p_u(u) = p_u(\sigma_1), s_2u \in \overline{\Pi_2} \text{ and }\( (s_1\sigma_1, s_2u) \in S),
ii) (\forall \sigma_2 \in \Sigma_2), s_2\sigma_2 \in \overline{\Pi_2} \text{ implies } (\exists v \in \Sigma_1^*), p_v(v) = p_v(\sigma_2), s_1v \in \overline{\Pi_1} \text{ and }\( (s_1v, s_2\sigma_2) \in S),
iii) s_1 \in H_1 \text{ implies } (\exists u \in \Sigma_2^*), p_u(u) = \epsilon, s_2u \in H_2 \text{ and }\( (s_1, s_2u) \in S),
iv) s_2 \in H_2 \text{ implies } (\exists v \in \Sigma_1^*), p_v(v) = \epsilon, s_1v \in H_1 \text{ and }\( (s_1v, s_2) \in S).
\]

We say that \( H_1 \) and \( H_2 \) are bisimilar with respect to \( \Sigma_s \) if there exists a bisimulation relation of \( H_1 \) and \( H_2 \) with respect to \( \Sigma_s \) satisfying the conditions listed in Definition 2, or \( \equiv_{H_1,H_2} \) is a bisimulation. Intuitively, two languages are bisimilar with respect to their shared events if a shared event is possible at some point in one language, the same shared event should also be possible in the other language and vice versa; and if a marked string is reached in one language, then from a corresponding string in the other language, a marked string is also reachable. The following definition for mutual controllability is necessary for the later discussion [7].
Definition 3. Let $\Sigma_1, \Sigma_2, \ldots, \Sigma_n$ be the event and $\Sigma := \bigcup_{i=1}^{n} \Sigma_i$, where $\Sigma_i \cap \Sigma_j \neq \emptyset$ for $i, j = \{1, 2, \ldots, n\}$ and $i \neq j$. It is also assumed that $\Sigma_i$ is split into controllable and uncontrollable events, i.e. $\Sigma_i = \Sigma_i^c \cup \Sigma_i^u$, and the control status of shared event of two subsystems agrees, i.e., $\Sigma_i^c \cap \Sigma_j = \Sigma_i^c \cap \Sigma_j^u$. Let $p_i$ be the natural projection from $\Sigma^*$ to $\Sigma_i^*$. Let $H_i \subseteq \Sigma_i^*$. We define that $H_i$ and $H_j$ are mutually controllable if

$$\forall \epsilon_i, \epsilon_j \subseteq \Sigma_i^*, \epsilon_j \subseteq \Sigma_j^*, \epsilon_j \cap (\Sigma_i^* \setminus H_i) = \emptyset$$

where $p_i^j$ and $p_j^i$ are the natural projections from $(\Sigma_i \cup \Sigma_j)^*$ to $\Sigma_i^*$ and $\Sigma_j^*$, respectively.

Roughly speaking, since $p_i^j (\epsilon_i)^{-1} (H_j)$ models the external behaviours of $G_j$, seen by $G_i$, two subsystems are mutually controllable if the closed behaviour of one subsystem is controllable with respect to the external behaviours of the other subsystem for the uncontrollable shared events and vice versa.

3. Problem Formulation. In this section, we formulate a problem of decentralised control of SCT. For a decentralised system, we assume that a centralised plant is a synchronous composition of several smaller decentralised plants [14]. The concurrent operations of several decentralised plants could achieve the optimal objectives that can be achieved by centralised plant under certain conditions [6]. The framework of decentralised control is as follows: we consider the case with two decentralised plant. Note that this can be extended easily to more general cases. Let $\Sigma_i$ be the event alphabet of a subplant $G_i$, for $i = 1, 2$. It is allowed that $\Sigma_1 \cap \Sigma_2 = \emptyset$. Assume that $\Sigma_i = \Sigma_i^c \cup \Sigma_i^u$. We assume that the control status of shared events are the same, i.e., $\Sigma_1 \cap \Sigma_2 = \Sigma_i \cap \Sigma_j$. Let $\Sigma := \Sigma_i \cup \Sigma_j$ be the event set of the global system, say $G$. Let $\Sigma_c := \Sigma_1 \cup \Sigma_2$, $\Sigma_u := \Sigma_1 \cap \Sigma_2$ and $\Sigma_s := \Sigma_1 \cap \Sigma_2$. Let $p_i$ and $p_u$ be the natural projections from $\Sigma^*$ to $\Sigma_i^*$ and to $\Sigma_s^*$, respectively. Let $L_{1,m}, L_2 \subseteq \Sigma_i^*$ represent respectively the marked and closed behaviours of system $G_i$. We assume that $L_i = L_{1,m}$. The marked and closed behaviours of the system $G$ are respectively $L_m = L_{1,m} \mid L_{2,m}$ and $L = L_1 \mid L_2$. Now let $E_i \subseteq L_{1,m}$ be the local specification language on the subsystem $G_i$, not necessarily prefix-closed. The overall specification is then $E := (p_{1|L})^{-1} (E_1) \cap (p_{2|L})^{-1} (E_2)$, where $(p_{1|L})$ denotes the restriction of $p_1$ on $L$. For a decentralised control approach, we synthesise local supervisors $S_i$ on $G_i$ whose closed-loop marked and closed behaviours are respectively $\kappa_{L_i}(E_1)$ and $\kappa_{L_i}(E_2)$. The closed behaviour under the concurrent supervisions of the local supervisors is then $(p_{1|L})^{-1} (\kappa_{L_1}(E_1)) \cap (p_{2|L})^{-1} (\kappa_{L_2}(E_2))$. For a centralised control approach, we can synthesise for the global specification $E$ and obtain $\kappa_{L}(E)$ as the closed behaviour of a global supervisor.

Now we establish the following problem.

Problem 1. For given $E_i \subseteq L_{1,m}$, where $i = 1, 2$, under what condition is it true that

$$(p_{1|L})^{-1} (\kappa_{L_1}(E_1)) \cap (p_{2|L})^{-1} (\kappa_{L_2}(E_2)) = \kappa_{L}(E)$$

and

$p_i (\kappa_{L_1}(E))$ is nonblocking with respect to $L_{1,m}$?
Problem 1 is to find conditions under which decentralised synthesis and control for local specifications can achieve the same optimality compared to the centralised synthesis, and the control of one decentralised plant never incur a blocking in other decentralised plants.

4. Decentralised control of DES using Bisimulation. The following theorem provides sufficient conditions for Problem 1.

Theorem 1. Let $\Sigma_i, \Sigma_{ic}, \Sigma_{iu}, L_{i,m}$ and $L_i$, for $i = 1, 2$, be given as in Section 3. Suppose that $L_1$ and $L_2$ are mutually controllable. Let $E_i \subseteq L_{i,m}$, for $i = 1, 2$. Suppose further that

i) $\kappa_{L_1}(E_1)$ and $\kappa_{L_2}(E_2)$ are bisimilar with respect to $\Sigma_s := \Sigma_1 \cup \Sigma_2$, for $i \neq j$

ii) $\kappa_{L_i}(E_i)$ and $L_{j,m}$ are bisimilar with respect to $\Sigma_s$.

Then problem 1 is solved.

Note that this theorem is different from the one in structural control [7]. As mentioned before, one of conditions given in [7] is too strong. In some applications, it is found that some unnecessary states need to be marked for the shared-event-marking condition to be satisfied. To relax this condition which is applied to the structure of the system, it is found to be necessary to seek conditions applied to the given specifications.

Also note that we consider a centralised system as the synchronous composition of decentralised systems. We obtain sufficient conditions for given local, not necessary prefix-closed, specification languages. However, again the conditions now become specification-dependent conditions. However the second condition (mutual controllability condition) is still applied into the structure. So whenever a specification is modified, only the condition using bisimulation concept needs to be verified, which is simpler to be done. Note that in here nonprefix-closed specification languages are allowed and hence the problem of blocking will be addressed, which is a more relaxed case compared to [8].

To prove Theorem 1, one requires the following lemmas and proposition.

Proposition 1. Let $L$ be a closed language over $\Sigma$. Let $E_i \subseteq L$ for $i = 1, 2, \cdots, n$. Suppose that $\{\kappa_L(E_i) \mid i = 1, 2, \cdots, n\}$ are nonconflicting. Then $\kappa_L(\bigcap_{i=1}^n E_i) = \bigcap_{i=1}^n \kappa_L(E_i)$.

Proof. See Appendix A.

Lemma 1. This lemma is for nonconflicting and nonblocking conditions with bisimulation concept. Let $H_i \subseteq \Sigma_i$ for $i = 1, 2$. Suppose that $H_1$ and $H_2$ are bisimilar with respect to $\Sigma_s := \Sigma_1 \cap \Sigma_2$ and $H := p_1^{-1}(H_1) \cap p_2^{-1}(H_2)$. Then

i) $p_1^{-1}(H_1)$ and $p_2^{-1}(H_2)$ are nonconflicting, i.e., $p_1^{-1}(H_1) \cap p_2^{-1}(H_2) = p_1^{-1}(H_1) \cap p_2^{-1}(H_2)$, and

ii) $p_i(H)$ is nonblocking with respect to $H_i$, i.e., $p_i(H) = p_i(H) \cap H_i$.

Proof. For nonconflicting, we will show only one inclusion since the other inclusion is always true. Let $s \in p_1^{-1}(H_1) \cap p_2^{-1}(H_2)$. Then $p_1(s) \in H_1$ and $p_2(s) \in H_2$.
Let \( s_1 := p_1(s) \) and \( s_2 := p_2(s) \). Then one has that \( p_s(s_1) = p_s p_1(s) = p_s(s) \) and \( p_s(s_2) = p_s p_2(s) = p_s(s) \). So, \( p_s(s_1) = p_s(s_2) \). Therefore by definition, one has that \( s_1 \equiv_{H_1, H_2} s_2 \). Since we assume that \( H_1 \) and \( H_2 \) are bisimilar, \( s_1 \equiv_{H_1, H_2} s_2 \) is a bisimulation. Also, since \( s_1 \in H_1 \), there is a string \( u \in \Sigma_1^+ \) such that \( s_1 u \in H_1 \). If \( u = \epsilon \), since \( s_1 \equiv_{H_1, H_2} s_2 \) there is a string \( t \in \Sigma_2^+ \) such that \( p_s(t) = \epsilon, s_2 t \in H_2 \) and \((s_1, s_2 t) \in \equiv_{H_1, H_2}\). Thus for a string \( st \in \Sigma^+ \), \( p_s(st) = p_s(s) \epsilon = s_1 u \in H_1 \) and \( p_2(st) = p_2(s) \epsilon = s_2 t \in H_2 \). Hence \( st \in p_1^{-1}(H_1) \cap p_2^{-1}(H_2) \). Therefore, 
\[
 s \in p_1^{-1}(H_1) \cap p_2^{-1}(H_2).
\]
Otherwise, write \( u = \sigma_1 \sigma_2 \cdots \sigma_n \), where \( \sigma_1, \sigma_2, \ldots, \sigma_n \in \Sigma_1 \) for some \( n \). Since \( s_1 \equiv_{H_1, H_2} s_2 \) and \( s_1 \sigma_1 \in H_1 \), there is a string \( v_1 \in \Sigma_2^+ \) such that \( p_s(v_1) = p_s(s_1) \), \( s_2 v_1 \in H_2 \), and \((s_1 \sigma_1, s_2 v_1) \in \equiv_{H_1, H_2}\). Repeating this for \( \sigma_2, \sigma_3, \ldots, \sigma_n \), one has 
\[
v_2, v_3, \ldots, v_n \in \Sigma_2^+ \text{ such that } p_s(v_i) = p_s(s_1 \sigma_1 \sigma_2 \cdots \sigma_i), s_2 v_1 v_2 \cdots v_i \in \equiv_{H_1, H_2},
\]
for \( i = 2, 3, \ldots, n \). Let \( v := v_1 v_2 \cdots v_n \in \Sigma_2^+ \). Since \( s_1 u \in H_1 \) and \((s_1 u, s_2 v) \in \equiv_{H_1, H_2}\), there is a string \( t \in \Sigma_2^+ \) such that \( p_s(t) = \epsilon, s_2 v t \in H_2 \) and \((s_1 u, s_2 v t) \in \equiv_{H_1, H_2}\). Thus \( s_1 u = p_1(s) u \in H_1 \) and \( s_2 v t = p_2(s) v t \in H_2 \), where \( u \in \Sigma_1^+, v \in \Sigma_2^+ \) and \( t \in (\Sigma_2 - \Sigma_1)^* \). For \( \sigma_i \in \Sigma_1 \cap \Sigma_2 \), write \( v_i = w_i \sigma_i r_i \), where \( w_i, r_i \in (\Sigma_2 - \Sigma_1)^* \).

Note that if \( \sigma_i \notin \Sigma_1 \cap \Sigma_2 \), i.e., \( \sigma_i \in \Sigma_1 - \Sigma_2 \), then \( p_s(\sigma_i) = \epsilon \). So, by Eq. (1), one has that \( p_s(v_i) = v, v_i \in (\Sigma_2 - \Sigma_1)^* \). Hence trivially we need to consider the case of \( \sigma_i \in \Sigma_1 \cap \Sigma_2 \). Let \( I_{\Sigma_1 \cap \Sigma_2} := \{ i \mid 1 \leq i \leq n, \sigma_i \in \Sigma_1 \cap \Sigma_2 \} \). That is, if \( i \in I_{\Sigma_1 \cap \Sigma_2} \), then \( v_i = w_i \sigma_i r_i \), where \( w_i, r_i \in (\Sigma_2 - \Sigma_1)^* \) and \( \sigma_i \in \Sigma_1 \cap \Sigma_2 \), and if \( i \notin I_{\Sigma_1 \cap \Sigma_2} \), then \( v_i \in (\Sigma_2 - \Sigma_1)^* \). Write \( I_{\Sigma_1 \cap \Sigma_2} = \{ i_1, i_2, \ldots, i_m \} \).

Let 
\[
x := \sigma_1 v_1 \cdots \sigma_{i_1-1} v_{i_1-1} \sigma_{i_1} v_{i_1+1} \sigma_{i_1+1} v_{i_1+2} \sigma_{i_2-1} v_{i_2-1} v_{i_2} \sigma_{i_2+1} v_{i_2+2} \cdots \sigma_{i_m-1} v_{i_m-1} v_{i_m} \sigma_{i_m+1} v_{i_m+2} \cdots
\]
\[
\cdots \sigma_n v_n t.
\]

Thus 
\[
p_1(sx) = p_1(s)p_1(x) = s_1 \sigma_1 \cdots \sigma_{i_1-1} \sigma_{i_1} \sigma_{i_1+1} \cdots \sigma_{i_2-1} \sigma_{i_2} \cdots \sigma_{i_m-1} \sigma_{i_m} \sigma_{i_m+1} \cdots \sigma_n = s_1 u \in H_1.
\]

Also, 
\[
p_2(sx) = p_2(s)p_2(x) = s_2 v_1 \cdots v_{i_1-1} v_{i_1+1} \cdots v_{i_2-1} v_{i_2+1} \cdots v_{i_m-1} v_{i_m+1} \cdots v_n t = s_2 v t \in H_2.
\]

Hence \( sx \in p_1^{-1}(H_1) \cap p_2^{-1}(H_2) \). Therefore, \( s \in p_1^{-1}(H_1) \cap p_2^{-1}(H_2) \).

Using a similar construction as the above, one can show that \( p_i(H) \) is nonblocking with respect to \( H_i \). \( \square \)

**Lemma 2.** Let \( H := \bigcap_{i=1}^n p_i^{-1}(H_i) \) and \( E_i \subseteq H_i \), for \( i = 1, 2, \ldots, n \). Suppose that \( p_1^{-1}(E_1), p_2^{-1}(E_2), \ldots, p_n^{-1}(E_n) \) are nonconflicting. Then \( (p_1|_H)^{-1}(E_1), (p_2|_H)^{-1}(E_2), \ldots, (p_n|_H)^{-1}(E_n) \) are nonconflicting.

**Proof.** See Appendix B. \( \square \)
Lemma 3. Let $\Sigma_1, \Sigma_i, \Sigma_u, L_{i,m}, L_i, E_i$ and $\kappa_{L_i}(E_i)$, for $i = 1, 2$, be given as in Section 3. Suppose that the conditions given in Theorem 1 hold. Then
\[(p_{1|L}^{-1}(\kappa_{L_i}(E_i))) = (p_{1|L})^{-1}(\kappa_{L_i}(E_i)).\]

Proof. See Appendix C.

Lemma 4. For given $E_i \subseteq L_{i,m}$, for $i = 1, 2$, suppose that the conditions given in Theorem 1 hold. Then
1. For all $\kappa_{L_i}(E_i) \in \mathcal{C}(L_i)$, $(p_{1|L})^{-1}(\kappa_{L_i}(E_i)) \in \mathcal{C}(L)$;
2. For all languages $K \in \mathcal{C}(L)$, $p_1(K) \in \mathcal{C}(L_i)$, where $\mathcal{C}(L)$ and $\mathcal{C}(L_i)$ are the sets of controllable sublanguages of $L$ and $L_i$.

Proof. To show (1), let $\kappa_{L_i}(E_i) \in \mathcal{C}(L_i)$ and $(\kappa_{L_i}(E_i))' := (p_{1|L})^{-1}(\kappa_{L_i}(E_i))$. To show that $(\kappa_{L_i}(E_i))'$ is controllable with respect to $L$, let $s \sigma \in (\kappa_{L_i}(E_i))' \Sigma_u \cap L$. Thus $s \in (\kappa_{L_i}(E_i))', \sigma \in \Sigma_u$, and $\sigma \in L$. So $s \in \mathcal{C}(E_i)(\kappa_{L_i}(E_i)) \subseteq (p_{1|L})^{-1}(\kappa_{L_i}(E_i))$, so $p_i(s) \in (\kappa_{L_i}(E_i))$. If $\sigma \in \Sigma_i$, then $p_i(s \sigma) = p_i(s) \sigma \in p_i(L) \subseteq L_i$. Since $\sigma \in \Sigma_i \cap \Sigma_u = \Sigma_{i,u}$ and $(\kappa_{L_i}(E_i)) \in \mathcal{C}(L_i)$,
\[p_i(s \sigma) = p_i(s) \sigma \in (\kappa_{L_i}(E_i))' \Sigma_u \cap L_i, \text{ or } s \sigma \in (p_{1|L})^{-1}(\kappa_{L_i}(E_i))).\]

Since $s \sigma \in L$, $s \sigma \in (p_{1|L})^{-1}(\kappa_{L_i}(E_i))) \cap L = (p_{1|L})^{-1}(\kappa_{L_i}(E_i))$. By Lemma 3 and the assumptions in Theorem 1, $\sigma \in (p_{1|L})^{-1}(\kappa_{L_i}(E_i)) = (p_{1|L})^{-1}(\kappa_{L_i}(E_i)) = (\kappa_{L_i}(E_i))'$. If $\sigma \notin \Sigma_i$, then $p_i(s \sigma) = p_i(s) \sigma \in (\kappa_{L_i}(E_i))$. By the same argument as above, $\sigma \in (\kappa_{L_i}(E_i))' \cap L$. Hence $(\kappa_{L_i}(E_i))' \cap \mathcal{C}(L)$.

To show (2), we show the case $i = 1$. Let $K \in \mathcal{C}(L)$ and $K' = p_1(K)$. To show $K' \in \mathcal{C}(L_1)$, let $\sigma \in K \Sigma_{i,u} \cap L_1$. Thus $s \in K', \sigma \in \Sigma_{i,u} \cap L_1$. So $s \in p_1(K) = p_1(K)$. There is a string $u \in K$ such that $s = p_1(u)$. Since $u \in K \subseteq L = \bigcap_{i=1}^{m} p_{1|L}(L_i)$, one has that $p_1(u) \in L_i$, for $i = 1, 2, \ldots, n$. Since $\sigma \in \Sigma_{i,u} \subseteq \Sigma_i$ and $p_1(u) = s$, $p_1(u \sigma) = p_1(u) \sigma = s \sigma \in L_1$, or $u \sigma \in p_{1|L}(L_1)$. For $j = 2, 3, \ldots, n$, if $\sigma \notin \Sigma_j$, $p_{j|L}(u \sigma) = p_j(u) \in L_j$, i.e., $u \sigma \in p_{j|L}(L_j)$; if $\sigma \in \Sigma_j$ then $s \sigma \in \Sigma_{i,u} \cap \Sigma_j$. Thus $p_j(u \sigma) = p_j(u) \sigma \in L_j \Sigma_{i,u} \cap \Sigma_j$. Since $\sigma \in (p_{j|L}(L_1))$, $p_{j|L}(u \sigma) \in p_j(p_{1|L}(L_1)) = (p_j)^{-1}(1))$. So by the mutual controllability of $L_1$ and $L_j$ (an assumption in Theorem 1), $p_{j|L}(u \sigma) \in L_j \Sigma_{i,u} \cap \Sigma_j \cap (p_{j|L}(L_1)) \subseteq L_j$. Thus $\sigma \in (p_{j|L}(L_1))$. Hence $\sigma \in \bigcap_{i=1}^{m} p_{1|L}(L_i) \subseteq L$. Since $\sigma \in \Sigma_{i,u} \cap L_i$, $K$ is controllable, $u \sigma \in K \Sigma_{i,u} \cap L \subseteq K$. Thus $p_1(u \sigma) = p_1(u) \sigma = s \sigma \in p_1(K) = p_1(K) = K'$. So $K' = p_1(K) \in \mathcal{C}(L_1)$.

Lemma 5. For given $E_i \subseteq L_{i,m}$, for $i = 1, 2$, suppose that the conditions given in Theorem 1 hold. Then for $\kappa_{L_i}(E_i) \in \mathcal{C}(L_i)$, $(p_{1|L})^{-1}(\kappa_{L_i}(E_i)) = \kappa_{L_i}(p_{1|L})^{-1}(E_i)$.

Proof. By (1) in Lemma 4, $(p_{1|L})^{-1}(\kappa_{L_i}(E_i)) \in \mathcal{C}(L)$. Since $(p_{1|L})^{-1}(\kappa_{L_i}(E_i)) \subseteq (p_{1|L})^{-1}(E_i), (p_{1|L})^{-1}(\kappa_{L_i}(E_i)) \subseteq (p_{1|L})^{-1}(E_i)$ for the reverse inclusion, since $\kappa_{L_i}(p_{1|L})^{-1}(E_i) \subseteq \mathcal{C}(L)$, by (2) in Lemma 4, $p_{1}(\kappa_{L_i}(p_{1|L})^{-1}(E_i)) \subseteq \mathcal{C}(L)$. Since $p_{i}(\kappa_{L_i}(p_{1|L})^{-1}(E_i)) \subseteq p_i((p_{1|L})^{-1}(E_i) \subseteq E_i, p_{i}(\kappa_{L_i}(p_{1|L})^{-1}(E_i)) \subseteq \kappa_{L_i}(E_i)$. Thus $\kappa_{L_i}(p_{1|L})^{-1}(E_i) \subseteq (p_{1|L})^{-1}(\kappa_{L_i}(E_i))$. Since $\kappa_{L_i}(p_{1|L})^{-1}(E_i) \subseteq L$, $\kappa_{L_i}(p_{1|L})^{-1}(E_i)$ is nonconflicting. Hence $p_{1|L}^{-1}(\kappa_{L_i}(E_i)) = \kappa_{L_i}(p_{2|L})^{-1}(E_i)$. Hence
κ_L((p_1|L)^{-1}(E_1)) and κ_L((p_2|L)^{-1}(E_2)) are nonconflicting. Now we can show that for \( i = 1, 2 \),

\[
\bigcap_i (p_i|L)^{-1}(κ_L(E_i)) = \bigcap_i (p_i|L)^{-1}(κ_L_i(E_i)) = \bigcap_i (κ_L(p_i|L)^{-1}(E_i)) = κ_L(\bigcap_i ((p_i|L)^{-1}(E_i))) = κ_L(\bigcap_i (κ_L_i(E_i))) = κ_L(E).
\]

This shows that if \( L_1 \) and \( L_2 \) are mutually controllable, \( E_i \subseteq L_{i,m} \), for \( i = 1, 2 \), and suppose further that

i) \( κ_L_i(E_i) \) and \( κ_L_j(E_j) \) are bisimilar with respect to \( Σ := Σ_1 ∪ Σ_2 \), for \( i ≠ j \)

ii) \( κ_L_i(E_i) \) and \( L_{j,m} \) are bisimilar with respect to \( Σ_i \),

then the concurrent actions of decentralised supervisors with local specifications will guarantee the optimal behaviour that could be achieved by the centralised supervisor with global specification without any conflicting and blocking problems.

5. Example. For a practical example, consider a simple chemical reactor as shown in Figure 1. The reactor is comprised of a feed valve \( V_1 \), a drain valve \( V_2 \), a weight measurement unit \( W_1 \), a timer \( T_1 \), and a continuous reaction controller \( C_1 \). For the decentralised control, the whole plant is divided into two subplants as shown in Figure 1: the filling-draining subplant (\( G_1 \)) and the reaction subplant (\( G_2 \)). Note that there are no shared components among subplants.

![Figure 1. A chemical batch reactor](image)

DES models for the components are presented in Figure 2(a) for \( G_1 \) and 2(b) for \( G_2 \). In the figure, a circle represents (\( ◦ \)) a state and an arrow (with a label) from an exit state to the entrance state represents an event. Also an entering arrow (\( −→ ◦ \)) indicates the initial state and an exiting arrow (\( ◦ −→ \)) represents the marker state,
while the double arrow (⊙ ←→) shows the initial state and the marker state. The arrow with a bar in the middle represents a controllable event. The synchronous composition of all elementary components generates the possible system behaviours.

![Figure 2(a). DES model for Plant 1(G₁)](image)

![Figure 2(b). DES model for Plant 2(G₂)](image)

The batch reaction process operates as follows. Firstly, the feed valve V₁ is opened (event α₁) to fill the reactor tank with material until W₁ indicates it is full (η₁). Then the feed valve is closed (β₁). The chemical reaction is then be enabled using reaction controller C₁ (γ). The duration of the reaction is timed by the timer T₁ which can be set for 3 minutes (µ₁), 5 minutes (µ₂) and 7 minutes (µ₃). Assume that we choose 5 minutes in this specific case. During the operation, the timer can be held at its current time (µ₄) if necessary and then it can be reset to the idle state (µ₅) or re-released (µ₆). When the set time is expired (uncontrollable event µ₇), the controller C₁ is disabled (δ) and the timer is reset (µ₈). Then the drain valve V₂ is opened (α₂) to discharge the finished product from the tank until W₁ indicates that the tank is empty (η₂). When this happens, the drain valve is closed (β₂). This completes one cycle of the batch reaction process. To enforce the process proceeds in a specified order, three controllable shared events (σ₁, σ₂, σ₃) are introduced to indicate a completion of a certain process: the event σ₁ represents that the filling process is completed and the event σ₂ represents the completion of the chemical reaction, while σ₃ represents that the draining is finished and indicates a complete cycle of the batch reaction. These controllable synchronised events are represented as flags and their DES models are shown in Figure 3(a) and 3(b).

The event sets for the decentralised plants are \( \Sigma_1 = \{\alpha_1, \alpha_2, \beta_1, \beta_2, \eta_1, \eta_2, \sigma_1, \sigma_2, \sigma_3\} \) and \( \Sigma_2 = \{\mu_1, \mu_2, \mu_3, \mu_4, \mu_5, \mu_6, \mu_7, \mu_8, \gamma, \delta, \sigma_1, \sigma_2, \sigma_3\} \). Their shared events

...
are $\Sigma_s = \{\sigma_1, \sigma_2, \sigma_3\}$ and all of them are controllable events. Due to this, the mutual controllability conditions are verified trivially. According to the operation procedure described above, the specifications in each decentralised plant can also be established. Their DES models are given in Figure 4(a) for $G_1$ and (b) for $G_2$.

It can be checked that $E_i \subseteq L_{i,m}$, for $i = 1, 2$.

![Figure 3. DES model for flags](image)

![Figure 4. DES model for specification $(E_i)$](image)

The supervisors are obtained using the algorithm [10] and their marked behaviours $\kappa_{L_i}(E_i)$ (Figure 5(a)) and $\kappa_{L_2}(E_2)$ (Figure 5(b)) are displayed in Figure 5. It can be checked that in this case, the conditions in Theorem 1 hold. So the concurrent actions of decentralised control will guarantee the optimality of the centralised operation. Note again that the mutual controllability condition is applied to the structure of the system while the conditions using bisimulation are applied to each specification. Hence if there is a minor change of specifications like the changes of sequences, only the conditions with bisimulation need to be verified again.

6. Conclusion. This paper presented a concurrent structural decentralised control framework using bisimulation concept as a way to relax the shared-event-marking condition of the structural decentralised control of discrete-event systems [7]. It has found that the conditions developed in this paper now become specification dependent, i.e., the condition needs to be checked for given specifications. So whenever a specification is modified, the conditions using bisimulation concept need to be verified, which is simpler to be done. The issue of how to establish these conditions as structural conditions, namely they hold for a set of specifications, is still an open question.

Appendix A. Proof of Proposition 1.

Since $\bigcap_{i=1}^{n} E_i \subseteq E_i$, one has that $\kappa_{L}(\bigcap_{i=1}^{n} E_i) \subseteq \kappa_{L}(E_i)$, for $i = 1, 2, \cdots, n$. Hence $\kappa_{L}(\bigcap_{i=1}^{n} E_i) \subseteq \bigcap_{i=1}^{n} \kappa_{L}(E_i)$. For the reverse inclusion, since $\mathbb{C}(L)$, the set of closed,
controllable sublanguages of \( L \), is a complete sublattice of \( \mathcal{P}(L) \) and \( \{\kappa_L(E_i)\mid i = 1, 2, \cdots, n\} \) are nonconflicting, \( \bigcap_{i=1}^{n} \kappa_L(E_i) = \bigcap_{i=1}^{n} \kappa_L(E_i) \in \mathcal{C}(L) \), where \( \mathcal{P}(L) \) is the power set of \( L \). Hence, \( \bigcap_{i=1}^{n} \kappa_L(E_i) \in \mathcal{C}(L) \). Also, since \( \bigcap_{i=1}^{n} \kappa_L(E_i) \subseteq \bigcap_{i=1}^{n} E_i \), one has that \( \bigcap_{i=1}^{n} \kappa_L(E_i) \subseteq \kappa_L(\bigcap_{i=1}^{n} E_i) \).

\[ \Delta \]

Figure 5. DES model for supervisors \((\kappa_L,(E_i))\)

**Appendix B.** Proof of Lemma 2.
As the other inclusion is obvious, we only need to show the following.
\[
\frac{(p_1|_{H})^{-1}(E_1) \cap (p_2|_{H})^{-1}(E_2) \cap \cdots \cap (p_n|_{H})^{-1}(E_n)}{\subseteq p_1^{-1}(E_1) \cap p_2^{-1}(E_2) \cap \cdots \cap p_n^{-1}(E_n)}
\]
\[
= p_1^{-1}(E_1) \cap p_2^{-1}(E_2) \cap \cdots \cap p_n^{-1}(E_n) \quad \text{(since they are nonconflicting)}
\]
\[
= (p_1|_{H})^{-1}(E_1) \cap (p_2|_{H})^{-1}(E_2) \cap \cdots \cap (p_n|_{H})^{-1}(E_n).
\]

\[ \Delta \]

**Appendix C.** Proof of Lemma 3.
We show the case \( i=1 \). As one inclusion is always true, we need to show the other inclusion. Due to the assumption in Theorem 1 and Lemma 1, we have
\[
(p_1|_{L})^{-1}(\kappa_L(E_1)) = p_1^{-1}(\kappa_L(E_1)) \cap L = p_1^{-1}(\kappa_L(E_1)) \cap p_1^{-1}(L_1) \cap p_2^{-1}(L_2)
\]
\[
= p_1^{-1}(\kappa_L(E_1) \cap L_1) \cap p_2^{-1}(L_2) = p_1^{-1}(\kappa_L(E_1)) \cap p_2^{-1}(L_{2,m})
\]
\[
= p_1^{-1}(\kappa_L(E_1)) \cap p_2^{-1}(L_{2,m}) = (p_1|_{L})^{-1}(\kappa_L(E_1)) \cap p_2^{-1}(L_{2,m})
\]
\[
= p_1^{-1}(\kappa_L(E_1)) \cap L_{i,m} \cap p_2^{-1}(L_{2,m})
\]
\[
= p_1^{-1}(\kappa_L(E_1)) \cap p_1^{-1}(L_{i,m}) \cap p_2^{-1}(L_{2,m})
\]
\[
= p_1^{-1}(\kappa_L(E_1)) \cap L_m \subseteq p_1^{-1}(\kappa_L(E_1)) \cap L = (p_1|_{L})^{-1}(\kappa_L(E_1)).
\]
REFERENCES

[1] G. Barrett and S. Lafortune, Bisimulation, the supervisory control problems and strong model matching for finite state machines, *Discrete-Event Dynamic Systems: theory and Applications*, 8 (1998), 377–429.

[2] K. Cai and W. M. Wonham, Supervisor localization: A top-down approach to distributed control of discrete-event systems, *IEEE Transaction on Automatic Control*, 55 (2010), 605–618.

[3] P. Gohari and W. M. Wonham, On the complexity of supervisory control design in the RW framework, *IEEE Transaction on Systems, Man and Cybernetics? Part B: Cybernetics*, 30 (2000), 643–652.

[4] R. C. Hill, J. R. Cury, M. H. de Queiroz, D. M. Tilbury and S. Lafortune, Multi-level hierarchical interface-based supervisory control, *Automatica*, 46 (2010), 1152–1164.

[5] J. Komenda and J. van Schuppen, Modular control of discrete-event systems with coalgebra, *IEEE transaction Automatic Control*, 53 (2008), 447–460.

[6] P. Kozak and W. M. Wonham, Fully decentralised solutions of supervisory control problems, *IEEE Transactions on Automatic Control*, 40 (1995), 2694–2697.

[7] S. H. Lee and K. C. Wong, Structural decentralised control of concurrent discrete-event systems, *European Journal of Control*, 8 (2002), 477–491.

[8] F. Lin and W. M. Wonham, Decentralised supervisory control of discrete-event systems, *Information Sciences*, 44 (1988), 199–224.

[9] R. Milner, *Communication and Concurrency*, Prentice Hall, NJ, USA, 1989.

[10] P. J. Ramadge and W. M. Wonham, Supervisory control of a class of discrete-event processes, *SIAM J. Control and Optimization*, 25 (1987), 206–230.

[11] P. J. Ramadge and W. M. Wonham, Modular supervisory control of discrete event systems, *Mathematics of Control, Signal and Systems*, 1 (1988), 13–30.

[12] O. Stursberg, Hierarchical and distributed discrete event control of manufacturing Processes, in the 17th *IEEE conference on emerging technologies and factory automation*, IEEE, 2012, 1–8.

[13] Y. Sun, H. Lin and B. M. Chen, Decentralised bisimilarity control of discrete event systems, in *Proceedings of 31st Chinese control conference*, IEEE, 2012, 2134–2139.

[14] Y. Willner and M. Heymann, Supervisory control of concurrent discrete-event systems, *International Journal of Control*, 54 (1991), 1143–1169.

[15] W. M. Wonham, Notes on Control of Discrete-Event Systems. ECE 1636F/1637S 2012-13, Department of Electrical Engineering, University of Toronto, Toronto, Canada, 2013.

[16] K. C. Wong and W. M. Wonham, Hierarchical control of discrete-event systems, *Discrete Event Dynamic System*, 6 (1996), 241–273.

[17] T. S. Yoo and S. Lafortune, A general architecture for decentralized supervisory control of discrete-event systems, *Discrete Event Dynamic System*, 12 (2002), 335–377.

[18] C. Zhou, R. Kumar and S. Jiang, Control of nondeterministic discrete-event systems for bisimulation equivalence, *IEEE Transactions on Automatic Control*, 51 (2006), 754–765.

[19] C. Zhou and R. Kumar, Bisimilarity enforcement for discrete event systems using deterministic control, *IEEE Transactions on Automatic Control*, 56 (2011), 2986–2991.

Received April 2015; 1st revision February 2016; final revision September 2016.

E-mail address: Sang-Heon.Lee@unisa.edu.au