Abstract: In this contribution, we consider structural decentralized DES and supplement the existing control architecture with a two-level hierarchy. For the proposed overall system, we prove hierarchical consistency and that the closed-loop behavior is nonblocking. A comprehensive example demonstrates the computational benefit of our method.

Keywords: supervisory control, structural decentralized control, hierarchical control.

1. INTRODUCTION

In the past decade a great variety of ideas have been studied to reduce the complexity of synthesis algorithms for the supervisory control of discrete event systems (Rudie and Wonham, 1992; Jiang et al., 2001; Yoo and Lafortune, 2000; Hubbard and Caines, 2002; da Cunha et al., 2002; Zhong and Wonham, 1990; Lee and Wong, 2002; Wong and Wonham, 1996). A key ingredient of promising approaches is to assume or to impose a particular control architecture, such that computationally expensive product compositions of individual subsystems can be either avoided altogether or at least postponed to a more favorable stage in the design process. Our contribution builds on two recent results from this category, namely structural decentralized control and hierarchical control.

The decentralized control architecture proposed by Lee and Wong (2002) addresses plant models that are composed of a number of subsystems, which are coupled via shared events. Specifications are given for each subsystem individually, and the task is to synthesize individual supervisors. As the subsystems are coupled, synthesis will in general need to refer to the synchronous product of all subsystems. Conditions under which such product can be avoided are given in Lee and Wong (2002).

In hierarchical architectures (Zhong and Wonham, 1990; da Cunha et al., 2002; Hubbard and Caines, 2002), controller synthesis is based on a plant abstraction (high-level model), which is supposed to be less complex than the original plant model (low-level model). Technically, abstractions can be defined as language projections. While from worst case scenarios projections are known to be of exponential computational complexity, application relevant cases with polynomial complexity are identified in Wong (1997). An important question is how to derive the plant abstraction, such that a high-level controller can be implemented by available low-level control actions (hierarchical consistency). A characterization of this property is given in Zhong and Wonham (1990).

In this paper, we consider the decentralized setting of Lee and Wong (2002), where the overall system is modelled by the synchronous product of the individual subsystems. As an abstraction, we propose the natural language projection based on the shared events. We then show that this abstraction does comply with hierarchical consistency as defined in Zhong and Wonham (1990). Furthermore, the high-level model can be computed by the synchronous product of the projections of the individual subsystems (rather than the projection of the product of the individual subsystems). Whenever the projections of the subsystems behave computationally nicely, this change of order
promises a substantial computational benefit. This is demonstrated by an example.

The outline of the paper is as follows. Basic notations and definitions of supervisory control theory are recalled in Section 2. Section 3 and Section 4 introduce structural decentralized and hierarchical control of discrete event systems, respectively. In Section 5, both methods are combined so as to form a decentralized and hierarchical control architecture. A comprehensive example in Section 6 illustrates our contribution.

2. PRELIMINARIES

We recall basic facts from supervisory control theory. (Wonham, 2001; Cassandras and Lafortune, 1999).

For a finite alphabet Σ, the set of all finite strings over Σ is denoted Σ*. We write s1s2 ∈ Σ* for the concatenation of two strings s1, s2 ∈ Σ*. We write s1 ≤ s when s1 is a prefix of s, i.e., if there exists a string s2 ∈ Σ* with s = s1s2. The empty string is denoted ε ∈ Σ*, i.e., sε = s for all s ∈ Σ*. A language over Σ is a subset H ⊆ Σ*. The prefix closure of H is defined by H := {s1 ∈ Σ* | ∃s ∈ H s.t. s1 ≤ s}. A language H is prefix closed if H = H.

The natural projection p1 : Σ* → Σ*, i = 1, 2, for the (not necessarily disjoint) union Σ = Σ1 ∪ Σ2 is defined iteratively: (1) let p1(ε) := ε, (2) for s ∈ Σ*, σ ∈ Σ, let p1(sσ) := p1(s)p1(σ) if σ ∈ Σi, or p1(sσ) := p1(s) otherwise. The set-valued inverse of p1 is denoted p1−1 : Σ* → 2Σ∗ : p1−1(t) := {s ∈ Σ∗ | p1(s) = t}. The synchronous product H1[|]|H2 ⊆ Σ* of two languages H1 ⊆ Σ1 and H2 ⊆ Σ2 is the partial language H = H1[|]|H2 := p1−1(H1) ∩ p2−1(H2) ⊆ Σ*.

A finite automaton is a tuple G = (X, Σ, δ, x0, Xm), where X is the finite set of states; Σ is the finite alphabet of events; δ : X × Σ → X is the partial transition function; x0 ∈ X is the initial state; and Xm ⊆ X is the set of marked states. We write δ(x, σ) if δ is defined at (x, σ). In order to extend δ to a partial function on X × Σ, recursively let δ(x, ε) := x and δ(x, σσ′) := δ(δ(x, σ), δ(x, σ′)). L(G) := {s ∈ Σ∗ | δ(x0, s)} and Lm(G) := {s ∈ L(G) : δ(x0, s) ∈ Xm} are the closed and marked language generated by the finite automaton G, respectively. For a formal definition of the synchronous composition of two automata G1 and G2 we refer to e.g. Cassandras and Lafortune (1999) and note that L(G1[|]|G2) = L(G1)[|]|L(G2).

When L(G) represents the plant behavior in a supervisory control context, we write Σ = Σc ∪ Σu, Σc ∩ Σu = ∅, to distinguish controllable (Σc) and uncontrollable (Σu) events. A control pattern is a set γ ⊆ Σc of Σc and the set of all control patterns is denoted Γ ⊆ 2Σc. A supervisor is a map S : L(G) → Γ, where S(s) represents the set of enabled events after the occurrence of string s; i.e., a supervisor can disable controllable events only. The language L(S/G) generated by G under supervision S is iteratively defined by (1) ε ∈ L(S/G) and (2) σσ ∈ L(S/G) if s ∈ L(S/G) and sσ ∈ L(G). Thus, L(S/G) represents the behavior of the closed-loop system. To take into account the marking of G, let Lm(S/G) := L(S/G) ∩ Σm.

The closed-loop system is nonblocking if Lm(S/G) = L(S/G), i.e., if each string in L(S/G) is the prefix of a marked string in Lm(S/G).

A language H is said to be controllable w.r.t. L(G) if there exists a supervisor S such that H = L(S/G).

The set of all languages that are controllable w.r.t. L(G) is denoted C(L(G)) and can be characterized by C(L(G)) = {H ⊆ L(G) | ∃S s.t. H = L(S/G)}. Furthermore, the set C(L(G)) is closed under arbitrary union. Hence, for every specification language E there uniquely exists a supremal controllable sublanguage of E w.r.t. L(G), which is formally defined as κ(E)(G) := ∪{K ∈ C(L(G)) | K ⊆ E}. A supervisor S that leads to a closed-loop behavior κ(E)(G) is said to be maximal permissive. A maximal permissive supervisor can be realized on the basis of a generator of κ(E)(G). The latter can be computed from G and a generator of E. The computational complexity is of order O(N2M2), where N and M are the number of states in G and the generator of E, respectively.

A language E is Lm-closed if E ∩ Σm = E and the set of Lm(G)-closed languages is denoted Fm(G). For specifications E ∈ Fm(G), the plant L(G) is nonblocking under maximal permissive supervision.

3. STRUCTURAL DECENTRALIZED DES

Structural decentralized DES as proposed in Lee and Wong (2002) are composed of subsystems, realized by finite state automata Gs, i = 1, 2, . . . , n with respective alphabets Σs. The synchronization of each two subsystems Gi and Gj is organized via shared events Σj ∩ Σj.

Definition 3.1. A decentralized control system (DCS) consists of subsystems, modelled by finite state automata Gi, i = 1, . . . , n over the respective alphabets Σi. The overall system is defined as G := ∏i=1n Gi over the alphabet Σ := ∪i=1n Σi. The controllable and uncontrollable events are Σc := Σ ∩ Σc and Σuc := Σ ∩ Σu, respectively, where Σc ∩ Σuc = Σc ∩ Σuc = Σc ∩ Σuc = Σc ∩ Σuc = ∅. For brevity and convenience, let L := L(G), Lm := Lm(G), Li := L(Gi), and Lm,i := Lm(Gi).

For our applications we assume local specifications Ei ∈ Fm,i ⊆ Σi, i = 1, . . . , n for each subsystem Gi. Relative to the overall alphabet Σ, the Ei become (pi)−1(Ei), where pi : Σi → Σ is the natural projection. Taking into account the language L of G, the global specification is E := ∪i=1n (pi)−1(Ei) ∩ L.

There are two approaches for generating a supervisor implementing the given set of specifications: the synthesis of a monolithic supervisor S for the overall specification E leading to the closed language L(S/G) = κ(E) of the supervised system, and the synthesis of
local supervisors $S_i$ for $G_i$ and $E_i$, $i = 1, \ldots, n$ resulting in $\|_{i=1}^{n} L(S_i/G_i) = \bigcap_{i=1}^{n} (p_i^{-1}(K_{L_i}(E_i))) \cap L$; for the second approach see Figure 1.

Fig. 1. Structural Decentralized Control Structure

In Lee and Wong (2002), conditions under which both approaches lead to the same overall result are developed. Thus, for $i = 1, \ldots, n$ the following holds:

$$\int_{i=1}^{n} (p_i^{-1}(K_{L_i}(E_i))) = \overline{K_{L_i}(E_i)}, \quad (1)$$

$$p_i(K_{L_i}(E_i)) \text{ is nonblocking w.r.t. } L_{i,m}. \quad (2)$$

**Theorem 3.1.** (Structural Decentralized DES Lee and Wong (2002)). Using notation as in Definition 3.1, denote the natural projection $p_i^{ij} : (\Sigma_i \cup \Sigma_j)^* \to \Sigma_i^*$; Suppose that $E_i \in \mathcal{F}_{i,u}$ and that for $i, j = 1, \ldots, n, i \neq j$

(i) $L_{i,m}$ marks $\Sigma_i \cap \Sigma_j$, i.e.

$$\Sigma_i^*(\Sigma_i \cap \Sigma_j) \cap L_{i,m} \subseteq L_{i,m}(\Sigma_i \cap \Sigma_j) \quad (3)$$

(ii) $L_i$ and $L_j$ are mutually controllable, i.e.

$$\mathcal{T}_i(\Sigma_i \cap \Sigma_j) \cap p_i^{ij}((p_j^{ij})^{-1}(\mathcal{T}_j)) \subseteq \mathcal{T}_i \quad (4)$$

Then (1) and (2) hold.

In order to discuss the computational complexity, let $N$ and $M$ denote bounds for the number of states in $G_i$ and generators of $E_i$, respectively. Then $N^N$ and $M^N$ are the respective bounds for the number of states in the overall system. The computational complexity of the monolithic synthesis procedure is $O(N^{2N}M^{2M})$, whereas it is $O(nN^N M^N)$ for the local synthesis approach. It is clear that this benefit does not come for free. The approach in Lee and Wong (2002) requires the computation of the language projection (with exponential worst case complexity) in condition (4). However, $G = \|_{i=1}^{n} G_i$ need not be computed.

### 4. HIERARCHICAL CONTROL

We consider the event-based hierarchical control scheme in Zhong and Wonham (1990) (Figure 2 (a)).

The detailed plant model $G$ and the supervisor $S^0$ form a low-level closed-loop system, indicated by $\text{Con}^{\text{ilo}}$ (control action) and $\text{In}^{\text{ilo}}$ (feedback information). Similarly, the high-level closed loop consists of an abstract plant model $G^h$ and the supervisor $S^h$. The two levels are interconnected via $\text{Con}^{\text{hilo}}$ and $\text{In}^{\text{hilo}}$.

The former allows $S^h$ to impose high-level control on $S^0$, the latter drives the abstract plant $G^h$ in accordance to the detailed model. From the perspective of the high-level supervisor, the forward path sequence $\text{Con}^{\text{hilo}}$, $\text{Con}^{\text{ilo}}$ is usually designated "command and control", while the feedback path sequence $\text{In}^{\text{hilo}}, \text{In}^{\text{ilo}}$ is identified with "report and advise". Formally, the high-level abstraction is defined as follows.

**Definition 4.1.** (Hierarchical Abstraction). Let $G = (X, \Sigma, \delta, x_0, X_m)$ be a finite automaton and $\Sigma^h \subseteq \Sigma$ a set of high-level events. A reporter map $p^h : \Sigma^h \to (\Sigma^h)^*$ such that (1) $\theta(e) = e$ and (2) either $\theta(s) = \theta(s^h)$ or $\theta(s) = \theta(s)^h$, where $s \in \Sigma$, $s^h \in \Sigma^h$. The high-level language is defined by $L^h := \theta(L(G))$. The high-level marking is chosen s.t. $L_m^h \subseteq L^h$, where $L_m^h$ is required to be regular. The canonical recognizer of $L_m^h$ is denoted $G^h$, and hence, $L(G^h) = L^h$. $L_m^h(G^h) = L_m^h$ Finally, high-level controllable and uncontrollable events are denoted $\Sigma_m^h$ and $\Sigma_u^h$, respectively, where $\Sigma^h = \Sigma^h \cap \Sigma_m^h$, $\Sigma^h \cap \Sigma_u^h = \emptyset$.

For our control architecture, not only the high-level events and the reporter map, but also the choice of controllable and uncontrollable events $\Sigma_m^h$ and $\Sigma_u^h$ are essential. For the synchronous composition of subsystems, we will show in Section 5 that this choice can be based on shared events.

Fig. 2. Hierarchical Control Schemes

For our further discussion, we need to define the interconnection of low- and high-level supervisors with the plant.

**Definition 4.2.** (Hierarchical Control System). Referring to the notation in Definition 4.1, a hierarchical control system (HCS) consists of $G$, $G^h$, $S^h$ and $S^0$, where the high-level supervisor $S^h$ and the low-level supervisor $S^0$ fulfill the following conditions: $S^h : L^h \to \Gamma^h$ with the high-level control patterns $\Gamma^h := \{s^h \subseteq \Sigma^h \subseteq \Sigma\}$; and $S^0 : L(G) \to \Gamma$ with $S^0 \subseteq \theta(L(S^0/G)) \subseteq L(G^h/G^0)$.

Given a high-level specification $E^h \in \mathcal{F}_{h^h}$, we can synthesize $S^h$ such that $LS^h(G^0) = \kappa_{h^h}(E^h)$ with a nonblocking high-level closed-loop. At this stage, the remaining task is to implement high-level control actions for the low-level plant by means of $S^0$.

---

1 Note also that in this case $E \in \mathcal{F}_{h^h}$.
2 In the sequel, we focus our attention on the reporter map $\theta = p^h$, where $p^h$ is the natural language projection into $(\Sigma^h)^*$.
Definition 4.3. (Hierarchical Control Problem). Given \( G, G^h_i, S^h_i \), find a low-level supervisor \( S^{lo} \) as in Definition 4.2 such that the low-level controlled language of the HCS, \( L(S^{lo}/G) \), is nonblocking.

On the one hand, there may not exist a low-level supervisor \( S^{lo} \) such that \( \theta(L(S^{lo}/G)) = L(S^h_i/G^h_i) \), and we end up with a strict subset relation. This is interpreted as an optimistic high-level supervisor expecting low-level behavior that is not possible. On the other hand, if the above equation turns out true for some \( S^{lo} \), it provides a powerful tool to show that the low-level is nonblocking. This is one motivation for the following notion of hierarchical consistency.

Definition 4.4. (Hierarchical Consistency (HC)). The hierarchical control system in Definition 4.2 is hierarchically consistent if the following applies:

\[
\theta(L(S^{lo}/G)) = L(S^h_i/G^h_i).
\]  

There has been a variety of event-based contributions which tackle the complexity problem associated with the hierarchical abstraction and which provide appropriate definitions of the high-level marking as well as high-level controllable events in order to prove hierarchical consistency. Zhong and Wonham (1990) introduces the notion of strict-output-control-consistency to guarantee a consistent low-level controller fulfilling a high-level specification. Blocking is not considered in this approach. da Cunha et al. (2002) utilizes a generalized model for controlled DES in the high-level to obtain hierarchical consistency without adding complexity by refining the hierarchy. Moor et al. (2003) uses Willems’ I/O behaviours to establish a nonblocking hierarchical control architecture. Other state-based approaches (Hubbard and Caines, 2002) use state-aggregation for hierarchical abstraction.

5. HIERARCHICAL CONTROL FOR DECENTRALIZED SYSTEMS

We discuss how a combination of the decentralized approach in Section 3 with the hierarchical approach in Section 4 can improve the computational efficiency of supervisor synthesis; see Figure 2 (b) for our proposed overall scheme. As our detailed plant model, we consider a structural decentralized DES \( G_1, \ldots, G_n \), such as local feedback control by supervisors \( S_1, \ldots, S_n \). The abstraction \( G^{hi} \) of the low-level closed loop is based on the observation of shared events \( S^h_i \) via \( Inf^{lo} \). A high-level supervisor \( S^{hi} \) is designed to control \( G^h_i \), where effect on the low level is taken via \( Com^{hi} \) and a low-level supervisor \( S^{lo} \).

Definition 5.1. A Hierarchical and Decentralized Control System (HDCS) consists of the following entities.

- A detailed plant model is a decentralized control system \( G := \{ |_{i=1}^n G_i \} \) with subsystems \( G_i \) over respective alphabets \( \Sigma_i, i = 1, \ldots, n \) and \( \Sigma := \bigcup_{i=1}^n \Sigma_i \). Relevant languages are \( L_i := L(G_i), L_m := L_m(G_i), L_c := L(G), \) and \( L_{cm} := L_m(G) \), control- and uncontrollable events are \( \Sigma_c = \Sigma \cap \Sigma_c \) and \( \Sigma_{1m} = \Sigma \cap \Sigma_{1m} \) as in Definition 3.1.

- Local low-level controllers are denoted \( S_i : L_i \rightarrow \Gamma_i \), where \( \Gamma_i \) are the respective control patterns. Low-level closed-loop languages are denoted \( \Sigma_i := L(S_i/G_i), L_{cm} := L_{cm}/\Gamma_{cm} \), \( L_c := |_{i=1}^n L_i \), \( L_m := |_{i=1}^n L_{cm} \). Let \( \mathbf{G} \) be a generator such that \( L_c = L(G) \).

- For the abstraction the reporter map: \( \theta := p^{hi} \) is used, where \( p^{hi} : \Sigma^* \rightarrow (\Sigma^h)^* \) denotes the natural projection and \( \Sigma^h := \bigcup_{i,j \neq j} (\Sigma_i \cap \Sigma_j) \). The high-level language is \( L^{hi} = \theta(L_c) \).

\[
\theta(L(S^{lo}/G)) = L(S^h_i/G^h_i).
\]  

Lemma 5.1. (High Level Plant). Assume the control architecture of Definition 5.1 and let \( L^{hi} = p^{hi}(L^c_i) \). Then the high-level language is \( L^{hi} = p^{hi}(|_{i=1}^n L^c_i) \).

Proof (Sketch): We use induction and the identity \( p_0(H_1||H_2) = p_0(H_1) \) for languages \( H_1 \) and \( H_2 \) with alphabets \( \Sigma_{H_1} \) and \( \Sigma_{H_2} \) and the natural projection \( p_0 : (\Sigma_{H_1} \cup \Sigma_{H_2})^* \rightarrow (\Sigma_0)^* \) with \( \Sigma_0 \subseteq (\Sigma_{H_1} \cup \Sigma_{H_2}) \) and \( (\Sigma_{H_1} \cup \Sigma_{H_2}) \subseteq \Sigma_0 \) as in Wonham (2001).

The previous lemma enables the computation of the hierarchical abstraction by only using the hierarchically abstracted subsystems and it is not necessary to compute the full low-level plant automaton. Consequently, the computational effort reduces from computing \( |_{i=1}^n L^c_i \) to computing \( p^{hi}(|_{i=1}^n L^c_i) \) and then evaluating \( |_{i=1}^n p^{hi}(L^c_i) \). While technically the computational complexity is of the same order, a significant computational benefit shows in applications where realizations of \( p^{hi} \) have less states than those of \( L^c_i \).

For our further discussion, we assume that the low-level subsystems fulfill conditions (i) and (ii) in Theorem 3.1 and that the low-level supervisors have been synthesized to enforce a local specification \( E_i \in FL^{hi} \), i.e., \( L^c_i : L(S_i/G_i) = \kappa_{E_i}(E_i) \). Then, as a result of Theorem 3.1, the low-level is nonblocking, i.e., \( L^c = F^{lo} \).

The next theorem gives a possible choice of the low-level supervisor such that the proposed control architecture is hierarchically consistent and nonblocking.

Theorem 5.1. (Main Result): Let the hierarchical control architecture for structural decentralized DES be

\[ \text{By construction } L^{lo} \text{ is regular.} \]
defined as in Definition 5.1 and define the low-level supervisor $S_0^L$ for each $s \in L_c$ as

$$S_0^L(s) := S_0^L(p_0(s)) \cup (\Sigma - \Sigma^H)$$

Then $S_0^L$ is valid according to Definition 4.2, the HDCS is hierarchically consistent and the controller is maximally permissive. In particular, $S_0^L$ solves the hierarchical control problem in Definition 4.3

**Proof:** We use induction to show that $S_0^L$ is valid, i.e. $p_0^L(L(S_0^L(G^L)) \subseteq L(S_0^L(G^H))$. Obviously $e \in L(S_0^L(G^L))$ and $p_0^L(e) \in L(S_0^L(G^H))$. Pick any $s$ and $p \in \Sigma$ with $s_0 \in L(S_0^L(G^L))$ and $p_0^L(s_0) \in L(S_0^L(G^H))$. Then either $s \in \Sigma^L$ or $s \in (\Sigma - \Sigma^H)$. In the first case, $s \in S^L(s)$ implies $s \in S^H(s_0(p(s)))$ and, hence, $p_0^L(s_0) = p_0^L(s_0(s)) \in L(S_0^H(G^H))$. In the second case $p_0^L(s) = p_0^L(s) \in L(S_0^H(G^H))$.

We prove hierarchical consistency by induction. As validity has been established above, we are left to show $p^L(L(S_0^L(G^L)) \subseteq \Sigma^L \cap L_c$. It is clear that $e \in p^L(L(S_0^L(G^L))$ and $e \in \Sigma^L \cap L_c$. Let $s^H \in \Sigma^L \cap L_c$ and $s^H \in p^H(L(S_0^L(G^L))$ and assume for $s^H \in \Sigma^L \cap L_c$.

Then $S^H \in S^H(S^H)$ and by definition of the low-level supervisor $S_0^L$, $\forall s \in (p_0^L - 1)(\Sigma^H) \cap L_c$ we know that $s_0 \in S_0^L(s)$. Further, on, as $j^H \in S^H(S^H)$ it follows that $\exists s^H \in L_c$ s.t. $s^H \in (p_0^L - 1)(\Sigma^H)$ and $s^H \in L_c$. But then $s^H \in L_c$ and thus $p_0^L(s^H) = p_0^L(s) = p_0^L(s^H) \in L(S_0^L(G^L))$.

For proving nonblocking behavior it must be shown that $L(S_0^L(G^L)) = L(S_0^L(G^L)) \cap L_c$. As $L(S_0^L(G^L)) \subseteq L(S_0^L(G^L)) \cap L_c$ is obvious, it is only $L(S_0^L(G^L)) \cap L_c$. Assume $s \in L(S_0^L(G^L))$ but $s \notin L(S_0^L(G^L)) \cap L_c$. As $L(S_0^L(G^L)) \subseteq L_c$, $s \in L_c$ holds, and thus $s \notin L(S_0^L(G^L)) \cap L_c$ requires that $\forall u \in \Sigma^L$ s.t. $su \in L_c$, $\Sigma^L \in \Sigma^H$ and $u^* \in \Sigma^L$. But we know that $\exists s^H \in L_c$ s.t. $u^* \in L_c$ and $s^H \in L(S_0^L(G^L))$. Now let $u$ be such that $su \in L_c$ and let $\Sigma^L \in \Sigma^H$ as above. Then it holds that $p_0(su') \in L_c$ for which $\Sigma^L \in \Sigma_c$ because of (3) and $\forall u \in \Sigma^L$. Let $i_1, \ldots, i_m$ be the corresponding indices, $\forall u \in (\Sigma - \Sigma^H)^L$ such that $p_0(su) \in L_c$. Thus $su^i_1u_2 \ldots u_m \in L^i_c$ and thus $s \notin L(S_0^L(G^L)) \cap L_c$.

For proving maximal permissiveness we assume that $\exists s_0^L$ s.t. $L(S_0^L(G^L)) \subseteq (p_0^L - 1)(L(S_0^H(G^H)))$ with $S_0^L$ from Theorem 5.3. Then $\exists s \in L(S_0^L(G^L))$ with $p_0^L(s) = L(S_0^H(G^H))$ but $s \notin L(S_0^L(G^L))$. Because of HC $\exists s^H \in L(S_0^H(G^H))$ with $p_0^L(s') = L(S_0^H(G^H))$ and $s = s' u$ with $u \in (\Sigma - \Sigma^H)^L$. But we know that $\forall u^* \in L(S_0^H(G^H))$, which contradicts the assumption and $S_0^L$ is maximally permissive.

By the above theorem, the complexity of synthesis for the high-level specification becomes $O(N^{2h}(M^{2h}))^2$ compared to $O(N^{2h}(M^{2h}))$ for the local monolithic synthesis; where $N$, $N^{hi}$, $M^{hi}$ denote number of states in $G^H$, $G^{hi}$ and the canonical recognizer of $E^{hi}$, respectively. Again, in the case of $N^{hi} < N$ we expect a computational benefit.

### 6. EXAMPLE

We consider the two cooperating machine cells $G_1$ and $G_2$ given in Figure 3. Both machines have 6 states.

Fig. 3. local machines $G_1$ and $G_2$

The event $\gamma$ represents the start of the cooperation of the two machines, $\delta$ indicates that the cooperation terminated successfully and $\varphi$ represents failure of the cooperation. The machines evolve independently while $a_1, \ldots, a_7$ or $b_1, \ldots, b_7$ occurs. Thus the shared events are $\Sigma_1 \cap \Sigma_2 = \{\gamma, \delta, \varphi\}$ and the uncontrollable shared events are $\{\delta, \varphi\}$.

#### 6.1 Hierarchical Method

Following the lines of Lee and Wong (2002) (3) holds as all states before shared events are marked. The mutual controllability condition (4) for $i, j = 1, 2$ is also valid. For example, from Figure 3 and Figure 4 it can readily be observed that $\Sigma_1$ is controllable w.r.t. $(p_2^L)^{-1}(p_1^L)$ and the event set $\Sigma_2 \cap \Sigma_1$.

Fig. 4. $(p_2^L)^{-1}(p_1^L)$

Thus $G = G_1 \parallel G_2$ constitutes a structural-decentralized control system and we identify $G$ as the low-level plant of a HDCS. The low-level specifications $E_1$ and $E_2$ in Figure 5 are controllable w.r.t. their respective sub-systems and are identical to the low-level controlled subsystems $G_1$ and $G_2$.

Now, in addition to low-level supervision, the high-level specification $E^{hi}$ in Figure 6 is implemented by employing our hierarchical control method. $E^{hi}$ states that after the occurrence of two successive failures in cooperation no more cooperation should take place.

---

4. This means all extensions of $s$ to a marked string must be disabled and disabling can only occur if a high-level event is disabled by $S_0^L$.

5. $\Phi = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7\}$. The corresponding transition is labeled with a tick as $\Phi$ contains controllable events.

6. The reverse relation is obvious because of symmetry.
Recent work in control of DES has been focused on exploiting structural properties for reducing the computational complexity of DES controller synthesis. Our approach embeds a decentralized approach Lee and Wong (2002) in a hierarchical control scheme and it was shown that our architecture guarantees nonblocking behavior of the controlled system. Furthermore we pointed out that hierarchical consistency need not be verified as it is directly implied by the proposed control architecture. The computational benefit of our method was illustrated by an example. Ongoing work aims for weaker conditions guaranteeing non-blocking behavior and the demonstration of our results by a laboratory case study of a manufacturing system.

7. CONCLUSIONS

REFERENCES