HIERARCHICAL CONTROL FOR STRUCTURAL DECENTRALIZED DES

Klaus Schmidt, Johann Reger, Thomas Moor

Lehrstuhl für Regelungstechnik, Universität Erlangen-Nürnberg
Cauerstraße 7, D-91058 Erlangen, Germany
klaus.schmidt@rt.eei.uni-erlangen.de

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 $s_1 s_2 \in \Sigma^*$ for the concatenation of two strings $s_1, s_2 \in \Sigma^*$. We write $s_1 \leq s$ when s_1 is a *prefix* of s, i.e. if there exists a string $s_2 \in \Sigma^*$ with $s = s_1 s_2$. The empty string is denoted $\varepsilon \in \Sigma^*$, i.e. $s\varepsilon = \varepsilon s = s$ for all $s \in \Sigma^*$. A *language* over Σ is a subset $H \subseteq \Sigma^*$. The *prefix closure* of H is defined by $\overline{H} := \{s_1 \in \Sigma^* | \exists s \in H \text{ s.t. } s_1 \leq s\}$. A language H is *prefix closed* if $H = \overline{H}$.

The *natural projection* $p_i : \Sigma^* \to \Sigma_i^*$, $i = 1, 2$, for the (not necessarily disjoint) union $\Sigma = \Sigma_1 \cup \Sigma_2$ is defined iteratively: (1) let $p_i(\varepsilon) := \varepsilon$; (2) for $s \in \Sigma^*$, $\sigma \in \Sigma$, let $p_i(s\sigma) := p_i(s)\sigma$ if $\sigma \in \Sigma_i$, or $p_i(s\sigma) := p_i(s)$ otherwise. The set-valued inverse of p_i is denoted $p_i^{-1} : \Sigma_i^* \to 2^{\Sigma^*}$, $p_i^{-1}(t) := \{s \in \Sigma^* | p_i(s) = t\}$. The *synchronous product* $H_1 || H_2 \subseteq \Sigma^*$ of two languages $H_i \subseteq \Sigma_i^*$ is $H_1 || H_2 = p_1^{-1}(H_1) \cap p_2^{-1}(H_2) \subseteq \Sigma^*$.

A *finite automaton* is a tuple $G = (X, \Sigma, \delta, x_0, X_m)$, where X is the finite set of *states*; Σ is the finite alphabet of *events*; $\delta : X \times \Sigma \to X$ is the partial *transition function*; $x_0 \in X$ is the *initial state*; and $X_m \subseteq X$ is the set of *marked states*. We write $\delta(x, \sigma)!$ if δ is defined at (x, σ). In order to extend δ to a partial function on $X \times \Sigma^*$, recursively let $\delta(x, \varepsilon) := x$ and $\delta(x, s\sigma) := \delta(\delta(x, s), \sigma)$, whenever both $x' = \delta(x, s)$ and $\delta(x', \sigma)!$. $L(G) := \{s \in \Sigma^* : \delta(x_0, s)!\}$ and $L_m(G) := \{s \in L(G) : \delta(x_0, s) \in X_m\}$ are the *closed* and *marked language* generated by the finite automaton G, respectively. For a formal definition of the synchronous composition of two automata G_1 and G_2 we refer to e.g. Cassandras and Lafortune (1999) and note that $L(G_1 || G_2) = L(G_1) || L(G_2)$.

When $L(G)$ represents the plant behavior in a supervisory control context, we write $\Sigma = \Sigma_c \cup \Sigma_u$, $\Sigma_c \cap \Sigma_u = \emptyset$, to distinguish *controllable* (Σ_c) and *uncontrollable* (Σ_u) events. A *control pattern* is a set γ, $\Sigma_u \subseteq \gamma \subseteq \Sigma$, and the set of all control patterns is denoted $\Gamma \subseteq 2^{\Sigma}$. A *supervisor* is a map $S\colon L(G) \to \Gamma$, 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) $\varepsilon \in L(S/G)$ and (2) $s\sigma \in L(S/G)$ iff $s \in L(S/G), \sigma \in S(s)$ and $s\sigma \in L(G)$. Thus, $L(S/G)$ represents the behavior of the *closed-loop system*. To take into account the marking of G, let $L_m(S/G) := L(S/G) \cap L_m(G)$. The closed-loop system is *nonblocking* if $\overline{L_m(S/G)} = L(S/G)$, i.e. if each string in $L(S/G)$ is the prefix of a marked string in $L_m(S/G)$.

A language H is said to be controllable w.r.t. $L(G)$ if there exists a supervisor S such that $\overline{H} = L(S/G)$. The set of all languages that are controllable w.r.t. $L(G)$ is denoted $\mathcal{C}(L(G))$ and can be characterized by $\mathcal{C}(L(G)) = \{H \subseteq L(G) |\ \exists S \text{ s.t. } \overline{H} = L(S/G)\}$. Furthermore, the set $\mathcal{C}(L(G))$ 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 $\kappa_{L(G)}(E) := \cup\{K \in \mathcal{C}(L(G)) |\ K \subseteq E\}$. A supervisor S that leads to a closed-loop behavior $\kappa_{L(G)}(E)$ is said to be *maximal permissive*. A maximal permissive supervisor can be realized on the basis of a generator of $\kappa_{L(G)}(E)$. The latter can be computed from G and a generator of E. The computational complexity is of order $O(N^2M^2)$, where N and M are the number of states in G and the generator of E, respectively.

A language E is L_m*-closed* if $\overline{E} \cap L_m = E$ and the set of $L_m(G)$-closed languages is denoted $\mathcal{F}_{L_m(G)}$. For specifications $E \in \mathcal{F}_{L_m(G)}$, 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 G_i, $i = 1, 2, \ldots, n$ with respective alphabets Σ_i. The synchronization of each two subsystems G_i and G_j is organized via shared events $\Sigma_i \cap \Sigma_j$.

Definition 3.1. A *decentralized control system* (DCS) consists of subsystems, modelled by finite state automata $G_i, i = 1, \ldots, n$ over the respective alphabets Σ_i. The overall system is defined as $G := ||_{i=1}^n G_i$ over the alphabet $\Sigma := \bigcup_{i=1}^n \Sigma_i$. The controllable and uncontrollable events are $\Sigma_{i,c} := \Sigma_i \cap \Sigma_c$ and $\Sigma_{i,u} := \Sigma_i \cap \Sigma_u$, respectively, where $\Sigma_c \cup \Sigma_u = \Sigma$ and $\Sigma_c \cap \Sigma_u = \emptyset$. For brevity and convenience, let $L := L(G)$, $L_m := L_m(G)$, $L_i := L(G_i)$, and $L_{i,m} := L_m(G_i)$.

For our applications we assume local specifications $E_i \in \mathcal{F}_{L_{i,m}} \subseteq \Sigma_i^*, i = 1, \ldots, n$ for each subsystem G_i. Relative to the overall alphabet Σ, the E_i become $(p_i)^{-1}(E_i)$, where $p_i : \Sigma^* \to \Sigma_i^*$ denotes the natural projection. Taking into account the language L of G, the global specification is $E = \bigcap_{i=1}^n (p_i)^{-1}(E_i) \cap L$.

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 $L(S/G) = \overline{\kappa_L(E)}$ of the supervised system, and the synthesis of local supervisors S_i for G_i and E_i, $i=1,\ldots,n$ resulting in $||_{i=1}^n L(S_i/G_i) = \bigcap_{i=1}^n (p_i)^{-1}\left(\overline{\kappa_{L_i}(E_i)}\right) \cap L$; for the second approach see Figure 1.

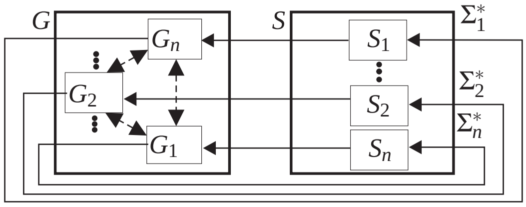

Fig. 1. Structural Decentralized Control Structure

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:

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

$$p_i(\overline{\kappa_L(E)}) \text{ is nonblocking w.r.t. } L_{i,m}. \tag{2}$$

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 $E_i \in \mathcal{F}_{L_{i,m}}$ and that for $i,j=1,\ldots,n$, $i \neq j$

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

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

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

$$\overline{L_i}(\Sigma_{j,u} \cap \Sigma_i) \cap p_i^{ij}((p_j^{ij})^{-1}(\overline{L_j})) \subseteq \overline{L_i} \tag{4}$$

Then (1) and (2) hold. [1]

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 $O(N^{2n}M^{2n})$, whereas it is $O(nN^2M^2)$ 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^{lo} form a low-level closed-loop system, indicated by Con^{lo} (control action) and Inf^{lo} (feedback information). Similarly, the high-level closed loop consists of an abstract plant model G^{hi} and the supervisor S^{hi}. The two levels are interconnected via Com^{hilo} and Inf^{lohi}. The former allows S^{hi} to impose high-level control on S^{lo}, the latter drives the abstract plant G^{hi} in accordance to the detailed model. From the perspective of the high-level supervisor, the forward path sequence Com^{hilo}, Con^{lo} is usually designated "command and control", while the feedback path sequence Inf^{lohi}, Inf^{hi} 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 $\Sigma^{hi} \subseteq \Sigma$ a set of high-level events. A *reporter map* [2] is a map $\theta : \Sigma^* \to (\Sigma^{hi})^*$ such that (1) $\theta(\varepsilon) = \varepsilon$ and (2) either $\theta(s\sigma) = \theta(s)$ or $\theta(s\sigma) = \theta(s)\sigma^{hi}$, where $\sigma \in \Sigma$, $\sigma^{hi} \in \Sigma^{hi}$. The high-level language is defined by $L^{hi} := \theta(L(G))$. The high-level marking is chosen s.t. $L_m^{hi} \subseteq L^{hi}$, where L_m^{hi} is required to be regular. The canonical recognizer of L_m^{hi} is denoted G^{hi}, and hence, $L(G^{hi}) = L^{hi}$, $L_m(G^{hi}) = L_m^{hi}$. Finally, high-level controllable and uncontrollable events are denoted Σ_c^{hi} and Σ_u^{hi}, respectively, where $\Sigma^{hi} = \Sigma_c^{hi} \cup \Sigma_u^{hi}$, $\Sigma_c^{hi} \cap \Sigma_u^{hi} = \emptyset$.

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

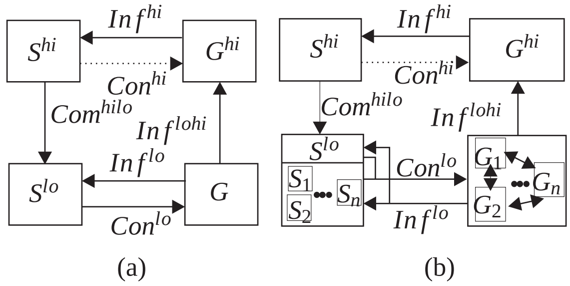

Fig. 2. Hierarchical Control Schemes

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^{hi}, S^{hi} and S^{lo}, where the *high-level supervisor* S^{hi} and the *low-level supervisor* S^{lo} fulfill the following conditions: $S^{hi}: L^{hi} \to \Gamma^{hi}$ with the high-level control patterns $\Gamma^{hi} := \{\gamma | \Sigma_u^{hi} \subseteq \gamma \subseteq \Sigma^{hi}\}$; and $S^{lo} : L(G) \to \Gamma$ with $\theta(L(S^{lo}/G)) \subseteq L(S^{hi}/G^{hi})$.

Given a high-level specification $E^{hi} \in \mathcal{F}_{L_m^{hi}}$, we can synthesize S^{hi} such that $L(S^{hi}/G^{hi}) = \kappa_{L^{hi}}(E^{hi})$ 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^{lo}.

[1] Note also that in this case $E \in \mathcal{F}_{L_m}$.

[2] In the sequel, we focus our attention on the reporter map $\theta = p^{hi}$, where p^{hi} is the natural language projection into $(\Sigma^{hi})^*$.

Definition 4.3. (Hierarchical Control Problem). Given G, G^{hi}, S^{hi}, 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 $\theta(L(S^{lo}/G)) = L(S^{hi}/G^{hi})$, and we end up with a strict subset relation. This is interpreted as an overoptimistic 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 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:

$$\theta(L(S^{lo}/G)) = L(S^{hi}/G^{hi}). \tag{5}$$

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 $G_1, \ldots G_n$, $G = ||_{i=1}^{n} G_i$, subject to 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 Σ^{hi} via Inf^{lohi}. A high-level supervisor S^{hi} is designed to control G^{hi}, where effect on the low level is taken via Com^{hilo} 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 Σ_i, $i = 1, \ldots, n$ and $\Sigma := \cup_{i=1}^{n} \Sigma_i$. Relevant languages are $L := L(G)$, $L_m := L_m(G)$, $L_i := L(G_i)$, and $L_{i,m} := L_m(G_i)$, controllable and uncontrollable events are $\Sigma_{i,c} = \Sigma_i \cap \Sigma_c$ and $\Sigma_{i,u} = \Sigma_i \cap \Sigma_u$ as in Definition 3.1.
- Local low-level controllers are denoted $S_i : L_i \to \Gamma_i$, where Γ_i are the respective control patterns. Low-level closed-loop languages are denoted $L_i^c := L(S_i/G_i)$, $L_{i,m}^c := L_i^c \cap L_{i,m}$, $L^c := ||_{i=1}^{n} L_i^c$, $L_m^c := ||_{i=1}^{n} L_{i,m}^c = L^c \cap L_m$. Also let G^c be a generator such that $L^c = L(G^c)$, $L_m^c = L_m(G^c)$.
- For the abstraction the reporter map: $\theta := p^{hi}$ is used, where $p^{hi} : \Sigma^* \to (\Sigma^{hi})^*$ denotes the natural projection and $\Sigma^{hi} := \bigcup_{i,j,i \neq j} (\Sigma_i \cap \Sigma_j)$. The high-level language is $L^{hi} := \theta(L^c)$ and [3]
$$L_m^{hi} := \{s^{hi} \in L^{hi} |\ (p^{hi})^{-1}(s^{hi}) \cap L_m^c \neq \emptyset\},$$
with canonical recognizer G^{hi} s.t. $L^{hi} = L(G^{hi})$, $L_m^{hi} = L_m(G^{hi})$. High-level controllable events are defined as $\Sigma_c^{hi} := \Sigma_c \cap \Sigma^{hi}$ and $\Sigma_u^{hi} := \Sigma_u \cap \Sigma^{hi}$.
- The high-level supervisor is denoted $S^{hi} : L^{hi} \to \Gamma^{hi}$ with the high-level closed-loop language $L(S^{hi}/G^{hi})$. A valid low-level supervisor $S^{lo} : L^c \to \Gamma$ must fulfill $\theta(L(S^{lo}/G^c)) \subseteq L(S^{hi}/G^{hi})$.

Lemma 5.1. (High Level Plant). Assume the control architecture of Definition 5.1 and let $L_i^{hi} = p^{hi}(L_i^c)$. Then the high level language is

$$L^{hi} = p^{hi}(||_{i=1}^{n} L_i^c) = ||_{i=1}^{n} L_i^{hi}.$$

Proof (Sketch): We use induction and the identity $p_0(H_1||H_2) = p_0(H_1)||p_0(H_2)$ for languages H_1 and H_2 with alphabets Σ_{H_1} and Σ_{H_2} and the natural projection $p_0 : (\Sigma_{H_1} \cup \Sigma_{H_2})^* \to (\Sigma_0)^*$ with $\Sigma_0 \subseteq (\Sigma_{H_1} \cup \Sigma_{H_2})$ and $(\Sigma_{H_1} \cap \Sigma_{H_2}) \subseteq \Sigma_0$ 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 $||_{i=1}^{n} L_i^c$ and subsequent projection $p^{hi}(||_{i=1}^{n} L_i^c)$ to first computing $p^{hi}(L_i^c)$ and then evaluating $||_{i=1}^{n} p^{hi}(L_i^c)$. While technically the computational complexity is of the same order, a significant computational benefit shows in applications where realizations of $p^{hi}(L_i^c)$ have less states than those of L_i^c.

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 $E_i \in \mathcal{F}_{L_{i,m}}$; i.e. $L_i^c := L(S_i/G_i) = \kappa_{L_i}(E_i)$. Thus, as a result of Theorem 3.1, the low-level is nonblocking, i.e. $L^c = \overline{L_m^c}$.

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

[3] By construction L_m^{hi} is regular.

defined as in Definition 5.1 and define the low-level supervisor S^{lo} for each $s \in L^c$ as

$$S^{lo}(s) := S^{hi}(p^{hi}(s)) \cup (\Sigma - \Sigma^{hi}).$$

Then S^{lo} is valid according to Definition 4.2, the HDCS is hierarchically consistent and the controller is maximally permissive. In particular, S^{lo} solves the hierarchical control problem in Definition 4.3

Proof: We use induction to show that S^{lo} is valid, i.e. $p^{hi}(L(S^{lo}/G^c)) \subseteq L(S^{hi}/G^{hi})$. Obviously $\varepsilon \in L(S^{lo}/G^c)$ and $p^{hi}(\varepsilon) = \varepsilon \in L(S^{hi}/G^{hi})$. Pick any s and $\sigma \in \Sigma$ with $s\sigma \in L(S^{lo}/G^c)$ and $p^{hi}(s) \in L(S^{hi}/G^{hi})$. Then either $\sigma \in \Sigma^{hi}$ or $\sigma \in (\Sigma - \Sigma^{hi})$. In the first case, $\sigma \in S^{lo}(s)$ implies $\sigma \in S^{hi}(p^{hi}(s))$ and, hence, $p^{hi}(s\sigma) = p^{hi}(s)\sigma \in L(S^{hi}/G^{hi})$. In the second case $p^{hi}(s\sigma) = p^{hi}(s) \in L(S^{hi}/G^{hi})$.

We prove hierarchical consistency by induction. As validity has been established above, we are left to show $p^{hi}(L(S^{lo}/G^c)) \supseteq \kappa_{L^{hi}}(E^{hi})$. It is clear that $\varepsilon \in p^{hi}(L(S^{lo}/G^c))$ and $\varepsilon \in \kappa_{L^{hi}}(E^{hi})$. Let $s^{hi} \in \kappa_{L^{hi}}(E^{hi})$ and $s^{hi} \in p^{hi}(L(S^{lo}/G^c))$ and assume for $\sigma^{hi} \in \Sigma^{hi}$, $s^{hi}\sigma^{hi} \in \kappa_{L^{hi}}(E^{hi})$. Then $\sigma^{hi} \in S^{hi}(s^{hi})$ and by definition of the low-level supervisor S^{lo}, $\forall s \in (p^{hi})^{-1}(s^{hi}) \cap L^c$ we know that $\sigma^{hi} \in S^{lo}(s)$. Further on, as $s^{hi}\sigma^{hi} \in L^{hi}$ it follows that $\exists s' \in L^c$ s.t. $s' \in (p^{hi})^{-1}(s^{hi})$ and $s'\sigma^{hi} \in L^c$. But then $s'\sigma^{hi} \in L(S^{lo}/G^c)$ and thus $p^{hi}(s'\sigma^{hi}) = p^{hi}(s')\sigma^{hi} = s^{hi}\sigma^{hi} \in p^{hi}(L(S^{lo}/G^c))$.

For proving nonblocking behavior it must be shown that $L(S^{lo}/G^c) = \overline{L(S^{lo}/G^c) \cap L^c_m}$. As $L(S^{lo}/G^c) \supseteq \overline{L(S^{lo}/G^c) \cap L^c_m}$ is obvious, we only prove $L(S^{lo}/G^c) \subseteq \overline{L(S^{lo}/G^c) \cap L^c_m}$. Assume $s \in L(S^{lo}/G^c)$ but $s \notin \overline{L(S^{lo}/G^c) \cap L^c_m}$. As $L(S^{lo}/G^c) \subseteq \overline{L^c_m}$, $s \in \overline{L^c_m}$ holds, and thus $s \notin \overline{L(S^{lo}/G^c) \cap L^c_m}$ requires that $\forall u \in \Sigma^*$ s.t. $su \in L^c_m$, $\exists \sigma^{hi} \in \Sigma^{hi}$ and $u', u'' \in \Sigma^*$ s.t. $u = u'\sigma^{hi}u''$ and $\sigma^{hi} \notin S^{lo}(su')$ and $\forall \overline{u'} \le u', s\overline{u'} \notin L^c_m$. [4] Now let u be such that $su \in L^c_m$ and let σ^{hi}, u', u'' as above. Then it holds that $p_i(su') \in L_{i,m}$ for i such that $\sigma^{hi} \in \Sigma_i$ because of (3) and $\forall i$ s.t. $\sigma^{hi} \notin \Sigma_i$ (let $i_1, \ldots, i_m$ be the corresponding indices), $\exists u_i \in (\Sigma_i - \Sigma^{hi})^*$ such that $p_i(s)u_i \in L^c_{i,m}$ as $L^c_i = \overline{L^c_{i,m}}$. Thus $su'u_{i_1}u_{i_2} \ldots u_{i_m} \in L^c_m$ and thus $s \in \overline{L(S^{lo}/G^c) \cap L^c_m}$.

For proving maximal permissiveness we assume that $\exists \tilde{S}^{lo}$ as in Definition 4.2 s.t. $L(S^{lo}/G^c) \subset L(\tilde{S}^{lo}/G^c) \subseteq (p^{hi})^{-1}(L(S^{hi}/G^{hi}))$ with S^{lo} from Theorem 5.1. Then $\exists s \in L(\tilde{S}^{lo}/G^c)$ with $s^{hi} = p^{hi}(s) \in L(S^{hi}/G^{hi})$ but $s \notin L(S^{lo}/G^c)$. Because of HC $\exists s' \in L(S^{lo}/G^c)$ with $p^{hi}(s') = s^{hi}$ and $s = s'u$ with $u \in (\Sigma - \Sigma^{hi})^*$. But we know that $(\forall s'u' < s'u), (\Sigma - \Sigma^{hi}) \in S^{lo}(su')$ and thus $s = s'u \in L(S^{lo}/G^c)$, which contradicts the assumption and S^{lo} is maximally permissive. □

By the above theorem, the complexity of synthesis for the high-level specification becomes $O((N^{hi})^{2n}(M^{hi})^2)$ compared to $O(N^{2n}(M^{hi})^2)$ for the local monolithic synthesis; where N, N^{hi}, M^{hi} denote number of states in G^c, 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.

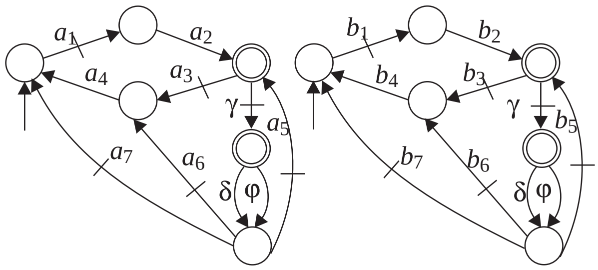

Fig. 3. local machines G_1 and G_2

The event γ represents the start of the cooperation of the two machines, δ indicates that the cooperation terminated successfully and φ represents failure of the cooperation. The machines evolve independently while $a_1, \ldots, a_7$ or $b_1, \ldots, b_7$ occur. 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 [5] it can readily be observed that $\overline{L_1}$ is controllable w.r.t. $(p^{12}_2)^{-1}(p^{12}_1(\overline{L_2}))$ and the event set $\Sigma_{2u} \cap \Sigma_1$. [6]

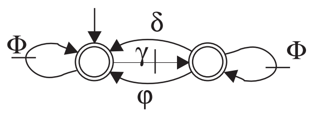

Fig. 4. $(p^{12}_2)^{-1}(p^{12}_1(\overline{L_{2,m}}))$

Thus $G = G_1 || G_2$ 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 subsystems and are identical to the low-level controlled subsystems G^c_1 and G^c_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^{lo}.

[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 Φ contains controllable events.

[6] The reverse relation is obvious because of symmetry.

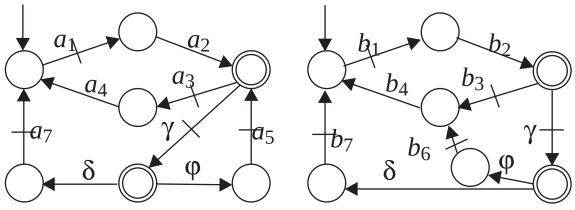

Fig. 5. low-level specifications E_1 and E_2

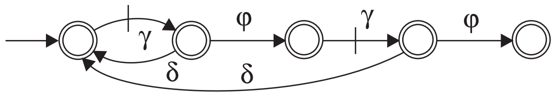

Fig. 6. high-level specification E^{hi}

For the abstraction, $\Sigma^{hi} = \{\gamma, \delta, \varphi\}$ and Figure 7 shows recognizers G_1^{hi} and G_2^{hi} of the respective supervised subsystems G_1^c and G_2^c. It turns out that, in this case, the computation of the projection is easy as the co-operating part and the independent part of the two machines are clearly separated. The high-level plants have only two states. By computing the synchronous composition of the abstracted subplants G_1^{hi} and G_2^{hi}, we obtain the two-state high-level plant $G^{hi} = G_1^{hi} \| G_2^{hi}$. It is evident that this is easier than computing $p^{hi}(L_1^c \| L_2^c)$, where $G_1^c \| G_2^c$ has 26 states.

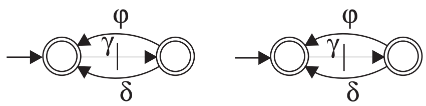

Fig. 7. high-level abstractions G_1^{hi} and G_2^{hi}

It can easily be verified that E^{hi} is controllable w.r.t. $L(G^{hi})$ and the high-level controlled language $L(S^{hi}/G^{hi})$ equals E^{hi} with a 5-state recognizer. Because of Theorem 5.1 the low-level supervisor $S^{lo}(s) = S^{hi}(p^{hi}(s)) \cup (\Sigma - \Sigma^{hi})$ realizes the desired high-level behavior and implements non-blocking and maximally permissive behavior in the low level.

6.2 Comparison with other methods

For the standard RW control synthesis the overall plant and specification automata have 26 states and 14 states, respectively and the supervised plant has 77 states. For applying the method of Zhong and Wonham (1990); da Cunha et al. (2002) it is also necessary to compute the overall plant automaton. The reporter map for the high-level abstraction is defined by labeling the low-level strings and for Zhong and Wonham (1990) there is additional computational effort for verifying hierarchical consistency, whereas in da Cunha et al. (2002), a supervisory control problem must be solved for each low-level subsystem, corresponding to a high-level string via the reporter map. For the technique of Lee and Wong (2002) the additional high-level specification E^{hi} can be decomposed into two local specifications $p_i(E^{hi})$, $i = 1, 2$ which leads to low-level specifications with 8 states, each, and supervised low-level plants with 17 states both, versus 7 states for the case without high-level control. Thus, compared to our method the high-level specification must be decomposed, which leads to relatively large low-level supervisors.

7. CONCLUSIONS

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 nonblocking behavior and the demonstration of our results by a laboratory case study of a manufacturing system.

REFERENCES

C.G Cassandras and S. Lafortune. Introduction to discrete event systems. *Kluwer*, 1999.

A.E.C. da Cunha, J.E.R. Cury, and B.H. Krogh. An assume guarantee reasoning for hierarchical coordination of discrete event systems. *WODES*, 2002.

P. Hubbard and P.E. Caines. Dynamical consistency in hierarchical supervisory control. *IEEE TAC*, 2002.

S. Jiang, V. Chandra, and R. Kumar. Decentralized control of discrete event systems with multiple local specializations. *Proc. ACC*, 2001.

S-H. Lee and K.C. Wong. Stuctural decentralised control of concurrent discrete-event systems. *EJC*, 2002.

T. Moor, J. Raisch, and J.M. Davoren. Admissibility criteria for a hierarchical design of hybrid control systems. In *Proc. ADHS'03*, 2003.

K. Rudie and W.M. Wonham. Think globally, act locally: decentralized supervisory control. *IEEE TAC*, 1992.

K. Wong. On the complexity of projections of discrete-event systems., 1997. URL `citeseer.nj.nec.com/wong98complexity.html`.

K.C. Wong and W.M. Wonham. Hierarchical control of discrete-event systems. *Discrete Event Dynamic Systems*, 1996.

W.M Wonham. Notes on control of discrete event systems. *Department of Electrical & Computer Engineering, University of Toronto*, 2001.

T. Yoo and S. Lafortune. A generalized framework for decentralized supervisory control of discrete event systems. *WODES*, 2000.

H. Zhong and W.M. Wonham. On the consistency of hierarchical supervision in discrete-event systems. *IEEE TAC*, 1990.