New Results on Decentralized Supervisory Control of Discrete-Event Systems

Tae-Sic Yoo and Stéphane Lafortune
Department of Electrical Engineering and Computer Science
The University of Michigan,
1301 Beal Avenue, Ann Arbor, MI 48109–2122, U.S.A.
{tyoo, stephane}@eecs.umich.edu; www.eecs.umich.edu/umdes

Abstract

We present new results on decentralized supervisory control of discrete-event systems. We generalize the control architecture by allowing combinations of "fusion by intersection" and "fusion by union" for the control actions issued by the individual (local) supervisors. The algebraic properties of co-observability in the context of this general architecture are presented. We show that appropriate combinations of fusion rules with corresponding decoupled local decision rules guarantee the safety of the closed-loop behavior with respect to a given specification that is not co-observable. We characterize an "optimal" combination of fusion rules among those combinations guaranteeing the safety of the closed-loop behavior. In addition, a simple supervisor synthesis technique generating the infimal prefix-closed controllable and co-observable superlanguage is presented.

1 Introduction

In the conventional "conjunctive" decentralized control architecture for discrete-event systems (conjunctive architecture for the sake of brevity), the control actions of a set of individual supervisors are combined (or "fused") using the *intersection* of locally *enabled* events. Most of the results on decentralized supervisory control are based on the conjunctive architecture.[1] An exception is the work in [5], where decentralized supervision with a fixed local decision rule under various decision fusions is considered. In [8], we present a generalized form of the conjunctive architecture where the control actions of a set of supervisors can be fused using both union and intersection of enabled events. Under this general architecture, the local supervisors decide *a priori* that some controllable events will be processed by "fusion by union" of enabled events and that other controllable events will be handled by "fusion by intersection" of enabled events. The most salient feature of this general architecture is that a larger class of languages can be achieved than before since a *relaxed version* of the notion of co-observability of [6] appears in the necessary and sufficient conditions for the existence of supervisors.

The objective of this paper is to investigate how to deal with the situation where the relaxed version of co-observability of [8] (called simply "co-observability" hereafter) fails to hold. The contributions of this paper are:

- The algebraic properties of co-observability are presented in Section 3. These properties show that the supremal and infimal co-observable elements of a class of languages need not exist, in general.
- A simple decentralized supervisor synthesis technique decoupling the synthesis of local supervisors is developed under the general architecture. Because of the intentional separation of the design of the local decision rules, this technique circumvents the difficulties caused by the dependency of local decision rules in the design of decentralized supervisors.
- Equipped with the above synthesis technique, we present in Section 4.2 a rule for partitioning the set of controllable events (between "fusion by union" and "fusion by intersection") that guarantees the safety of the closed-loop behavior with respect to a given specification that is not co-observable. We also characterize an "optimal" partitioning rule of the set of controllable events among the partitions guaranteeing the safety of the closed-loop behavior.
- Several properties of the closed-loop behavior generated by the above synthesis technique are presented in Section 4.3.
- We also present a simple synthesis procedure of the supervisor that results in the infimal prefix-closed controllable and co-observable (in the sense of [6]) superlanguage in Section 4.3. The local supervisors are separately synthesized in a direct manner that avoids the explicit computation of the infimal prefix-closed controllable and co-observable superlanguage.

General knowledge of supervisory control and its most common notations are assumed. For introductory materials the reader is directed to [1]. Due to space limitations, the proofs of some results have been omitted from

[1] For references on decentralized supervisory control with the conjunctive architecture, we refer the reader to [7].

this paper; detailed discussions and proofs are available in [7].

2 Review of the General Architecture

We start by recalling some key definitions and results of [8] that are needed before we can present our new results (Sections 3 and 4).

2.1 The General Architecture

The problem of decentralized supervision can be described as follows. Each local supervisor i, $i \in \{1, \ldots, n\}$, has its own sensors (observable event set $\Sigma_{o,i}$) and control authorities (controllable event set $\Sigma_{c,i}$). Collectively, the supervisors observe $\Sigma_o = \Sigma_{o,1} \cup \ldots \cup \Sigma_{o,n}$ and control $\Sigma_c = \Sigma_{c,1} \cup \ldots \cup \Sigma_{c,n}$. We denote by $\Sigma_{uo} = \Sigma \setminus \Sigma_o$ and $\Sigma_{uc} = \Sigma \setminus \Sigma_c$, the unobservable and uncontrollable event sets, respectively. *A priori* information available to each local site includes the uncontrolled system behavior, the desired system behavior, the sets $\Sigma_{c,i}$ and $\Sigma_{o,i}$, $i \in \{1, \ldots, n\}$, and the decision fusion rule employed to form a global control action on the system. The "conventional" decentralized architecture employs the conjunctive fusion rule for enabled events, thus requiring unanimous enabling of local decisions for global enablement; we call this rule "fusion by intersection" (of enabled events). Formally, a local decision rule is a function $S_{P_i} : P_i(\Sigma^*) \to \Gamma := \{\gamma \in 2^\Sigma : \Sigma_{uc} \subseteq \gamma\}$, where P_i is the "natural projection": $P_i : \Sigma \to \Sigma_{o,i}$ (see, e.g., [1]). The *conjunctive supervisor*, denoted by $\wedge_i S_{P_i}$, is defined as follows: $\wedge_i S_{P_i}(s) := \cap_{i=1}^n S_{P_i}(P_i(s))$.

Consider an architecture where "fusion by union" (of enabled events) is employed; let us call the resulting (global) supervisor a *disjunctive supervisor*. The disjunctive supervisor $\vee_i S_{P_i}$ is defined as follows: $\vee_i S_{P_i}(s) := \cup_{i=1}^n S_{P_i}(P_i(s))$. For the conjunctive architecture, the default policy for controllable events at the local supervisor is *enablement* and local supervisors directly disable the locally controllable events; this is called the "pass the buck" policy in [6]. Clearly, the *disablement* default should be employed by local supervisors for controllable events in the context of the disjunctive architecture, as local supervisors directly enable the locally controllable events.

A general architecture that combines "fusion by union" and "fusion by intersection" is introduced in [8]. The set of controllable events, Σ_c, is partitioned into $\Sigma_{c,e}$ and $\Sigma_{c,d}$: $\Sigma_c = \Sigma_{c,e} \dot\cup \Sigma_{c,d}$. $\Sigma_{c,e}$ is the set of controllable events for which the default setting is enablement while $\Sigma_{c,d}$ is the set of controllable events for which the default setting is disablement. The local decisions over $\Sigma_{c,e}$ are processed by the *conjunctive* fusion rule while the local decisions over $\Sigma_{c,d}$ are processed by the *disjunctive* fusion rule. Figure 1 is a conceptual diagram of the general architecture. Let us define a generalized

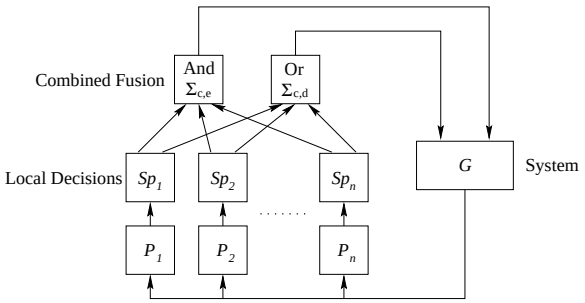

Figure 1: The general architecture

decentralized supervisor (*generalized supervisor* for the sake of brevity) with such a combined fusion rule, denoted by S_{gdec}, as

$$S_{gdec}(s) := P_{\Sigma_{c,e}}[\wedge_i S_{P_i}(s)] \cup P_{\Sigma_{c,d}}[\vee_i S_{P_i}(s)] \cup \Sigma_{uc},$$

where $P_{\Sigma_{c,e}}$ and $P_{\Sigma_{c,d}}$ are natural projection mappings: $P_{\Sigma_{c,e}} : \Sigma \to \Sigma_{c,e}$ and $P_{\Sigma_{c,d}} : \Sigma \to \Sigma_{c,d}$. The prefix-closed and marked languages generated by the general architecture can be defined in the usual manner [7].

2.2 Existence Result and Comparison Between the Architectures

As was mentioned above, in prior work on the conjunctive architecture, the default control action for a supervisor under insufficient information is to "enable" an event. We will refer to this default policy as the *permissive* local decision rule [8].

In the conjunctive architecture, co-observability is a key property for the existence of a set of local supervisors that achieve a given desired language. This property was introduced in [6]. We use an equivalent definition that is presented in [1]. Since we shall present a relaxed version of "co-observability" later on, we rename the conventional notion of co-observability as *C&P co-observability*. C refers to the "conjunctive" fusion rule for controllable events, while P refers to the "permissive" local decision rule: A language $K \subseteq M = \overline{M}$ is said to be C&P co-observable w.r.t. M, $\Sigma_{o,1}$, $\Sigma_{c,1}$, $\Sigma_{o,2}$, $\Sigma_{c,2}$,. . ., $\Sigma_{o,n}$, $\Sigma_{c,n}$, if $\forall s \in \overline{K}$ and $\forall \sigma \in \Sigma_c = \cup_{i=1}^n \Sigma_{c,i}$ s.t. $s\sigma \in M \setminus \overline{K}$, $\exists i \in \{1, \ldots, n\}$ s.t. $[P_i^{-1} P_i(s)\sigma \cap \overline{K} = \emptyset] \wedge [\sigma \in \Sigma_{c,i}]$.

Next, we recall the definition of *D&A co-observability* for the disjunctive architecture, the analogue of C&P co-observability for the conjunctive architecture, that was introduced in [8]. The term "D&A" stands for disjunctive and antipermissive. The reason for this term is that this property is for the *disjunctive* architecture and the *antipermissive* decision strategy should be applied at local supervisors. We say that the decision strategy at a local supervisor is "antipermissive" if the default control action for controllable events is disablement. The intuitive meaning of the antipermissive rule is to permit the occurrence of a controllable continuation only if the local supervisor has sufficient infor-

mation to determine with certainty that enabling the controllable event will not cause any violation of legality, namely, will not cause the closed-loop behavior to exceed the desired behavior.

Definition 1 *A language $K \subseteq M = \overline{M}$ is said to be D&A co-observable w.r.t. M, $\Sigma_{o,1}$, $\Sigma_{c,1}$,..., $\Sigma_{o,n}$, $\Sigma_{c,n}$, if $\forall s \in \overline{K}$ and $\forall \sigma \in \Sigma_c = \cup_{i=1}^{n} \Sigma_{c,i}$ s.t. $s\sigma \in \overline{K}$, $\exists i \in \{1,\dots n\}$ s.t. $[(P_i^{-1}P_i(s) \cap \overline{K})\sigma \cap M \subseteq \overline{K}] \wedge [\sigma \in \Sigma_{c,i}]$.*

Assume that we are given a partition of the set of controllable events $\Sigma_c = \Sigma_{c,e} \dot\cup \Sigma_{c,d}$. Let us define the following sets of events: For $i \in \{1,\dots n\}$, $\Sigma_{c,e,i} := \Sigma_{c,i} \cap \Sigma_{c,e}$ and $\Sigma_{c,d,i} := \Sigma_{c,i} \cap \Sigma_{c,d}$. $\Sigma_{c,e,i}$ is the set of locally controllable events whose default setting is enablement while $\Sigma_{c,d,i}$ is the set of locally controllable events whose default setting is disablement. We generalize C&P and D&A co-observability to embrace this partitioning of Σ_c; for the sake of simplicity, we call this generalized notion "co-observability".

Definition 2 *A language $K \subseteq M = \overline{M}$ is said to be co-observable w.r.t. M, $\Sigma_{o,1}$, $\Sigma_{c,d,1}$, $\Sigma_{c,e,1}$, $\Sigma_{o,2}$, $\Sigma_{c,d,2}$, $\Sigma_{c,e,2}$,..., $\Sigma_{o,n}$, $\Sigma_{c,d,n}$, $\Sigma_{c,e,n}$, if*
1. *K is C&P co-observable w.r.t. M, $\Sigma_{o,1}$, $\Sigma_{c,e,1}$,..., $\Sigma_{o,n}$, $\Sigma_{c,e,n}$.*
2. *K is D&A co-observable w.r.t. M, $\Sigma_{o,1}$, $\Sigma_{c,d,1}$,..., $\Sigma_{o,n}$, $\Sigma_{c,d,n}$.*

With this generalized notion of co-observability, the existence result of a decentralized supervisor achieving exactly a given desired behavior in the context of the general architecture, presented in [8], is as follows:

Theorem 1 *Consider a language $K \subseteq \mathcal{L}_m(G)$ where $K \neq \emptyset$ and consider a fixed partition of Σ_c such that $\Sigma_c = \Sigma_{c,d} \dot\cup \Sigma_{c,e}$. There exists a nonblocking generalized supervisor S_{gdec} such that $\mathcal{L}_m(S_{gdec}/G) = K$ and $\mathcal{L}(S_{gdec}/G) = \overline{K}$ iff the three following conditions hold:*
1. *K is controllable w.r.t. $\mathcal{L}(G)$ and Σ_{uc}.*
2. *K is co-observable w.r.t. $\mathcal{L}(G), \Sigma_{o,1}, \Sigma_{c,d,1}, \Sigma_{c,e,1}, \dots, \Sigma_{o,n}, \Sigma_{c,d,n}, \Sigma_{c,e,n}$.*
3. *K is $\mathcal{L}_m(G)$-closed.*

For a language satisfying the conditions in Theorem 1, the following local decision rules should be applied: For $s \in \Sigma^*$, let us define $E_i(s) = P_i^{-1}P_i(s) \cap \overline{K}$, $i \in \{1,\dots,n\}$. The local decision rules are:

$$\begin{aligned} &S_{P_i}(P_i(s)) = \\ &\{\sigma \in \Sigma_{c,d,i} : E_i(s)\sigma \cap \mathcal{L}(G) \subseteq \overline{K}\} \ \cup \\ &\{\sigma \in \Sigma_{c,e,i} : E_i(s)\sigma \cap \overline{K} \neq \emptyset\} \cup \\ &\Sigma_{c,e} \setminus \Sigma_{c,e,i} \ \cup \ \Sigma_{uc}. \end{aligned} \tag{1}$$

A detailed explanation of the intuitive meaning of this decision rule can be found in [7].

Let us define the following classes of languages where M is assumed to be prefix-closed and $K \subseteq M$:

$$\begin{aligned} \mathcal{L}_{cen}(K) &= \{L \subseteq K : L \text{ is observable w.r.t. } M, \Sigma_o, \Sigma_c\}, \\ \mathcal{L}_{DA}(K) &= \{L \subseteq K : L \text{ is D\&A co-observable w.r.t. } \\ &\quad M, \Sigma_{o,1}, \Sigma_{c,1}, \Sigma_{o,2}, \Sigma_{c,2}\}, \\ \mathcal{L}_{CP}(K) &= \{L \subseteq K : L \text{ is C\&P co-observable w.r.t. } \\ &\quad M, \Sigma_{o,1}, \Sigma_{c,1}, \Sigma_{o,2}, \Sigma_{c,2}\}, \\ \mathcal{L}_{gdec}(K) &= \{L \subseteq K : \exists \Sigma_{c,d} \text{ and } \Sigma_{c,e} \text{ s.t. } \Sigma_{c,d} \dot\cup \Sigma_{c,e} = \\ &\quad \Sigma_c \text{ and } L \text{ is co-observable w.r.t. } M, \Sigma_{o,1}, \\ &\quad \Sigma_{c,d,1}, \Sigma_{c,e,1}, \Sigma_{o,2}, \Sigma_{c,d,2}, \Sigma_{c,e,2}\}. \end{aligned}$$

Since controllability of the desired language is a required condition for the existence of supervisors among all architectures, the classes of languages defined above determine the performance (i.e., the class of achievable languages) of the architectures. The relations between these classes of languages are summarized in Fig. 2. Examples that justify the Venn diagram in Fig. 2 can be found in [7].

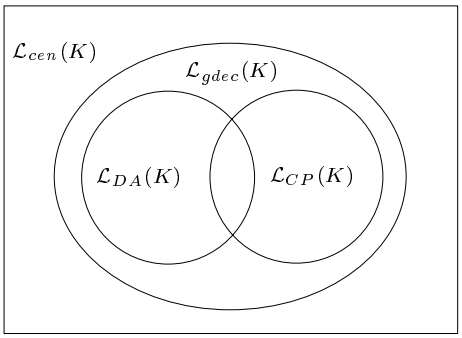

Figure 2: Performance comparison between the architectures

2.3 Test for Co-observability

Assume that H and G are trim finite-state automata generating the desired and uncontrolled languages K and $\mathcal{L}(G)$, respectively. In [6], the finite-state automaton M is constructed to test the C&P co-observability of $\mathcal{L}_m(H)$ in polynomial time. In [8], the finite-state automaton M_d, which is similar to M, is built to verify the D&A co-observability of $\mathcal{L}_m(H)$ in polynomial time. For further arguments, we define the set of terminal events:

$$\Sigma_{ter}(K) := \{\sigma : s\sigma \in K\}.$$

$\Sigma_{ter}(K)$ collects the events terminating the traces in K. The following theorem presented in [8] provides an algorithm to search for a partition of Σ_c satisfying co-observability in polynomial time.

Theorem 2 *There exist $\Sigma_{c,e}$ and $\Sigma_{c,d}$, a partition of Σ_c, such that $\mathcal{L}_m(H)$ is co-observable w.r.t. $\mathcal{L}(G)$, $\Sigma_{o,1}$, $\Sigma_{c,d,1}$, $\Sigma_{c,e,1}$, $\Sigma_{o,2}$, $\Sigma_{c,d,2}$, $\Sigma_{c,e,2}$, iff $\Sigma_{ter}(\mathcal{L}_m(M)) \cap \Sigma_{ter}(\mathcal{L}_m(M_d)) = \emptyset$.*

3 Algebraic Properties of Co-observability

Supremal sublanguages (with respect to a given property) play a key role in supervisory control problems,

since they provide the least restrictive solutions when the desired languages are not achievable under control (because they do not possess the property under consideration). Infimal superlanguages are also important (see, e.g., the range problem in [4], [6]). The existence of supremal and infimal languages can be verified by checking the closure under algebraic operations (union and intersection) of classes of languages. We present two algebraic properties of co-observability that are relevant in this regard.

Property 1 *D&A co-observability and co-observability are not preserved under union of languages.*

It is shown in [6] that C&P co-observability is preserved under intersection when prefix-closed languages are considered. However, the following property shows that this is not true for D&A co-observability.

Property 2 *D&A co-observability and co-observability are not preserved under intersection of languages.*

Properties 1 and 2 are disappointing in the sense that the notion of co-observability, which is key to the generalized architecture of Fig. 1, does not enjoy any of the algebraic properties that would be useful to address supervisor synthesis problems for desired behaviors that are not co-observable. For this reason, a different approach is necessary to tackle such problems. The next section presents new results in this regard.

4 Synthesis Results

4.1 Implementation of Generalized Supervisors

Let us recall the local decision rule (1). One can see that the local decision rules are decoupled from each other even though they work together in the context of the generalized architecture of Fig. 1. In addition to the decoupling of the local decision rules, the *information state* ($E_i(s) := P_i^{-1}P_i(s) \cap \overline{K}$) used in (1) is also independent of the decision rule (1). These observations lead us to propose the following approach to supervisor synthesis. Assume that the automaton describing the desired language is a strict subautomaton of the automaton generating the uncontrolled language. When the desired language is achievable (namely, it satisfies the conditions in Theorem 1), it is possible to design the estimator and control actions sequentially. That is, we can:

1. Build the local observers (estimators) of the automaton corresponding to the desired language.
2. Find the local control action for each local observer state according to the decision rule (1).

Moreover, supervision can be conducted in an *on-line* manner, if so desired. The next local observer state of the desired language can be found on-line upon the occurrence of a locally observable event and the local decision for this local observer state can be calculated once the local observer state is known. Note that the computation of the new observer state only requires the previous observer state and the current control action can be computed based on the current observer state only.

4.2 Design of Safe Generalized Supervisors

When the desired language is *not* achievable, one may want to synthesize a *safe* supervisor that guarantees that the closed-loop behavior stays within the desired language. We call this the inclusion problem and define it as follows:

(**Inclusion Problem**) Given uncontrolled system G over the set of events Σ and legal language K, find a nontrivial supervisor S such that $\mathcal{L}(S/G) \subseteq \overline{K}$.

For the perfect observation case ($\Sigma = \Sigma_o$), the supremal controllable sublanguage of the desired language K, $(\overline{K})^{\uparrow(C)}$, is computable and provides the least restrictive solution to the inclusion problem. Due to the lack of existence of supremal observable sublanguages, several approaches have been developed for control under imperfect observation. For centralized architectures, the property of normality has been suggested in order to compute a "suboptimal" solution to the inclusion problem. To improve upon this solution, other safe supervisor synthesis techniques were developed. Most of the efforts [2], [3] were devoted to the centralized architecture. There are very few results on the synthesis of safe supervisors in the context of decentralized architectures. One of the obstacles to the design of safe decentralized supervisors may be the mutual dependency of local decisions. To circumvent the dependency of local decisions, an intentional "decoupling" of the design of the local decision rules was suggested in [5]. The idea is to design local supervisors separately by following the *antipermissive rule* and fuse them through various fusion rules. Even though the performance of the supervisor may be degraded due to the separation of local supervisor design, the simplicity of this approach circumvents the mutual dependency of local decision rules. However, a drawback of the approach in [5] is that local supervisors do not exploit the structure of the fusion rule that is *a priori* known to each local supervisor. In the approach that we propose, we also decouple the design of local supervisors. However, the fusion rule and the local decision rules are accounted for in this design in order to enrich the closed-loop behavior.

Assume that the desired language K is controllable.[2] Let us denote by S_{gdec} the supervisor obtained following the decision rule in (1) with a given partition $\Sigma_{c,e}$ and

[2] This assumption is not restrictive since it is always possible to find the supremal element, $(\overline{K})^{\uparrow(C)}$. Moreover, it is not possible to find any closed-loop language which is in between $\overline{K}$ and $(\overline{K})^{\uparrow(C)}$ due to the supremality of $(\overline{K})^{\uparrow(C)}$.

$\Sigma_{c,d}$. In some sense, this means that S_{gdec} *pretends* that K is controllable *and co-observable*. We also build the automaton M and determine $\Sigma_{ter}(\mathcal{L}_m(M))$. Since the controllable events in $\Sigma_{ter}(\mathcal{L}_m(M))$ may cause a violation of safety (illegal continuation) if we follow the permissive rule for $\Sigma_{ter}(\mathcal{L}_m(M))$, we should use the antipermissive rule for these events.

The nature of the permissive rule is to enable when there is insufficient information. This rule can cause a violation of safety unless other local supervisors disable the events that would lead to illegal behavior. In contrast, the antipermissive rule disables when there is insufficient information. This conservative approach prevents the closed-loop behavior from being illegal. We have the following theorem providing a procedure for the synthesis of a safe supervisor under the general architecture.

Theorem 3 *If $\Sigma_{ter}(\mathcal{L}_m(M)) \subseteq \Sigma_{c,d}$, then synthesizing S_{gdec} according to (1) leads to $\mathcal{L}(S_{gdec}/G) \subseteq \overline{K}$.*

That is, every partition of the set of controllable events satisfying $\Sigma_{ter}(\mathcal{L}_m(M)) \subseteq \Sigma_{c,d}$, together with the corresponding local decision rules given in (1), guarantees a safe closed-loop behavior when S_{gdec} controls G, even if K is not co-observable.

In view of Theorem 3, one may wish to compare the closed-loop behaviors corresponding to different partitions. Let $\Sigma^1_{c,d}$, $\Sigma^2_{c,d}$, $\Sigma^1_{c,e}$, $\Sigma^2_{c,e} \in 2^{\Sigma_c}$. Suppose that $\Sigma_{ter}(\mathcal{L}_m(M)) \subseteq \Sigma^1_{c,d}, \Sigma^2_{c,d}$ and the following partition conditions hold: $\Sigma^1_{c,d} \dot\cup \Sigma^1_{c,e} = \Sigma_c$, $\Sigma^2_{c,d} \dot\cup \Sigma^2_{c,e} = \Sigma_c$. Let us denote the generalized supervisors following the supervision rule (1) with the above two partitions by S^1_{gdec} and S^2_{gdec}, respectively. Then we have the following theorem demonstrating the monotonicity of the closed-loop behaviors w.r.t. partitions.

Theorem 4 *If $\Sigma_{ter}(\mathcal{L}_m(M)) \subseteq \Sigma^2_{c,d} \subseteq \Sigma^1_{c,d}$, then $\mathcal{L}(S^1_{gdec}/G) \subseteq \mathcal{L}(S^2_{gdec}/G)$.*

Combining Theorems 3 and 4 leads to the following result.

Corollary 1 *If $\Sigma_{ter}(\mathcal{L}_m(M)) \subseteq \Sigma^2_{c,d} \subseteq \Sigma^1_{c,d}$, $\mathcal{L}(S^1_{gdec}/G) \subseteq \mathcal{L}(S^2_{gdec}/G) \subseteq \overline{K}$.*

One may be tempted to infer that the condition $\Sigma_{ter}(\mathcal{L}_m(M)) \subseteq \Sigma_{c,d}$ is necessary for safety. However, the following result demonstrates that it is not the case.

Proposition 1 *$\mathcal{L}(S_{gdec}/G) \subseteq \overline{K}$ does not imply $\Sigma_{ter}(\mathcal{L}_m(M)) \subseteq \Sigma_{c,d}$.*

However, the condition $\Sigma_{ter}(\mathcal{L}_m(M)) \subseteq \Sigma_{c,d}$ is critical to guarantee the safety of the closed-loop behavior.

Proposition 2 *If $\Sigma_{ter}(\mathcal{L}_m(M)) \not\subseteq \Sigma_{c,d}$, then the safety of the closed-loop behavior cannot be guaranteed.*

In view of the above results, we conclude that $\Sigma_{ter}(\mathcal{L}_m(M)) = \Sigma_{c,d}$ is the "optimal" partition, in the sense that it generates the largest safe closed-loop behavior among all the partitions guaranteeing the safety of the closed-loop behavior. Roughly speaking, the intuition behind Theorem 4 is that as the local decisions become "aggressive" (even though their fusion rule is conservative),[3] the closed-loop behavior becomes larger. Corollary 1 and Proposition 2 set a limit on "how aggressive" local decisions can be in order to guarantee the safety of the closed-loop behavior.

4.3 Properties of the Synthesized Language

Consider S_{gdec} defined in Section 4.2 with a partition of Σ_c guaranteeing the safety of the closed-loop behavior. Since the local decision rules are decoupled intentionally, it is natural to expect that the closed-loop behavior may not be maximally permissive when S_{gdec} is used. We have the following result.

Property 3 *(Non-maximality) In general, $\mathcal{L}(S_{gdec}/G)$ with the optimal partitioning is not a maximal controllable and co-observable sublanguage of $\overline{K}$.*

Since the local decision rule (1) does not consider marking, the following can be demonstrated as well.

Property 4 *(Blocking) In general, $\overline{\mathcal{L}_m(S_{gdec}/G)} \neq \mathcal{L}(S_{gdec}/G)$.*

In the remainder of this section, we consider the properties of two versions of S_{gdec} where the local decision rules are either always antipermissive or always permissive.

A language K is called strongly decomposable if

$$[P_1^{-1}P_1(\overline{K}) \cup P_2^{-1}P_2(\overline{K}) \cup \ldots \cup P_n^{-1}P_n(\overline{K})] \cap L(G) = \overline{K}.$$

This property has been considered as a decentralized version of normality in the sense that it is preserved under union and is a stronger condition than C&P co-observability [6]. Since controllability is also preserved under union, the supremal controllable and strongly decomposable sublanguage, denoted by $K^{\uparrow(CSD)}$, exists. Let us denote by $K^{\uparrow(CNi)}$ the supremal controllable and normal sublanguage of K, where controllability is w.r.t. $\mathcal{L}(G)$ and Σ_{uc} and normality is w.r.t. $\mathcal{L}(G)$ and $\Sigma_{o,i}$ (hence the superscript "i" in $K^{\uparrow(CNi)}$). We have the following result.

Property 5 *For all i, $K^{\uparrow(CSD)} \subseteq K^{\uparrow(CNi)}$.*

Let us set $\Sigma_c = \Sigma_{c,d}$ and denote the supervisor following the decision rule (1) with this partition as S^{ap}_{gdec}. This is

[3] Note that "aggressive" (permissive) local decision rules are matched with intersection (conservative fusion) and that "conservative" (antipermissive) local decision rules are matched with union (aggressive fusion).

the most conservative partition according to Corollary 1 and every local decision is based on the antipermissive local decision rule (hence the superscript "*ap*" in S^{ap}_{gdec}). We have the following inclusion.

Property 6 *For all i, $(\overline{K})^{\uparrow(CNi)} \subseteq \mathcal{L}(S^{ap}_{gdec}/G)$.*

In [5], the inclusions $(\overline{K})^{\uparrow(CSD)} \subseteq \mathcal{L}(S^{ap}_{gdec}/G) \subseteq \overline{K}$ were proved. Note that $(\overline{K})^{\uparrow(CNi)}$ is a suboptimal solution where all control authorities are given to the i-th local supervisor.[4] From Properties 5 and 6, we see that $(\overline{K})^{\uparrow(CNi)}$ provides a tighter lower bound than $(\overline{K})^{\uparrow(CSD)}$. Moreover, Corollary 1 states that $\mathcal{L}(S^{ap}_{gdec}/G)$ is the most conservative language guaranteed to be safe given that the decoupled local control actions following the decision rule (1) are applied.

We conclude this section with one last result when the local decision rule is always permissive. A formula for the infimal prefix-closed controllable and C&P co-observable superlanguage is known [6]. One of the purposes of the computation of this infimal superlanguage, denoted by $(\overline{K})^{\downarrow(CC\&P)}$, is to implement a supervisor generating this language. We present a simple algorithm synthesizing directly the decoupled local supervisors that result in this language without having to explicitly compute the infimal prefix-closed controllable and C&P co-observable superlanguage. We set $\Sigma_{c,e} = \Sigma_c$ and denote the supervisor following the decision rule (1) with this partition as S^{p}_{gdec}. This implies the architecture is the conjunctive one and every decision is based on the permissive decision rule. Then we have the following theorem. Note that the controllability assumption on K is not needed for this theorem.

Theorem 5 $\mathcal{L}(S^{p}_{gdec}/G) = (\overline{K})^{\downarrow(CC\&P)}$.

Theorem 5 can be interpreted as another characterization of $(\overline{K})^{\downarrow(CC\&P)}$, namely, as the closed-loop behavior that results from S^{p}_{gdec}. The discussion in Section 4.1 is directly applicable for the implementation of the local supervisors. By adding self-loops for enabled unobservable events at each estimator state, the automata representing the local supervisors can be realized. The automaton generating $(\overline{K})^{\downarrow(CC\&P)}$ can be constructed by forming the product of these automata with the system model.

5 Conclusion

We have presented new results on supervisor synthesis in the context of the general decentralized control architecture for discrete-event systems shown in Fig. 1. Building on the results in [8], we have presented simple "decoupled" control policies for the local supervisors and studied their properties. The design of these local supervisors is carried out as if the local supervisors were capable of achieving the desired behavior, namely, as if the desired language were co-observable. Under this technique, we found the "optimal" partition of the set of controllable events guaranteeing the safety of the closed-loop behavior. This simple supervisor synthesis technique can also be applied to synthesize local supervisors generating the infimal prefix-closed controllable and C&P co-observable superlanguage.

[4] Note that the control authorities are distributed throughout the local supervisors under decentralization. The purpose of Property 6 is to illustrate the properties of the language $\mathcal{L}(S^{ap}_{gdec}/G)$.

Acknowledgment

This research is supported in part by the DDR&E MURI on Low Energy Electronics Design for Mobile Platforms and managed by ARO under grant ARO DAAH04-96-1-0377.

References

[1] C. G. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems.* Kluwer Academic Publishers, 1999.

[2] N. B. Hadj-Alouane, S. Lafortune, and F. Lin. Centralized and distributed algorithm for on-line synthesis of maximal control policies under partial observation. *Discrete Event Dynamical Systems: Theory and Applications*, 6(41):379–427, 1996.

[3] M. Heymann and F. Lin. On-line control of partially observed discrete event systems. *Discrete Event Dynamical Systems: Theory and Applications*, 4(3):221–236, July 1994.

[4] F. Lin and W. M. Wonham. On observability of discrete-event systems. *Information Sciences*, 44(3):173–198, 1988.

[5] J. H. Prosser, M. Kam, and H. G. Kwatny. Decision fusion and supervisor synthesis in decentralized discrete-event systems. In *Proc. 1997 Ameri. Contr. Conf.*, pages 2251–2255, June 1997.

[6] K. Rudie and W. M. Wonham. Think globally, act locally: Decentralized supervisory control. *IEEE Trans. on Automat. Contr.*, 37(11):1692–1708, November 1992.

[7] T. Yoo and S. Lafortune. A general architecture for decentralized supervisory control of discrete-event systems. Technical report, Univ. of Michigan, 2000.

[8] T. Yoo and S. Lafortune. A general architecture for decentralized supervisory control of discrete-event systems. In *Discrete Event Systems: Analysis and Control*, pages 111–118. Kluwer Academic Publishers, 2000.