Computation of Controllable and Coobservable Sublanguages in Decentralized Supervisory Control via Communication

Jan Komenda* and Tomáš Masopust**

*Institute of Mathematics, Czech Academy of Sciences, Žižkova 22, 616 62 Brno, Czech Republic, komenda@ipm.cz

**Institute of Theoretical Computer Science and Center of Advancing Electronics Dresden (cfaed), TU Dresden, Germany, tomas.masopust@tu-dresden.de

Abstract

In decentralized supervisory control, several local supervisors cooperate to accomplish a common goal (specification). Controllability and coobservability are the key conditions to achieve a specification in the controlled system. We construct a controllable and coobservable sublanguage of the specification by using additional communications between supervisors. Namely, we extend observable events of local supervisors via communication and apply a fully decentralized computation of local supervisors. Coobservability is then guaranteed by construction. Sufficient conditions to achieve the centralized optimal solution are discussed. Our approach can be used for both prefix-closed and non-prefix-closed specifications.

Index terms— Discrete-event systems Decentralized supervisory control Coobservability Separability Communication

1 Introduction

Supervisory control theory of discrete-event systems (DES) modeled by finite automata was introduced by Ramadge and Wonham [31]. It aims to guarantee that the control specification consisting of safety and/or nonblockingness is satisfied in the controlled system. Supervisory control is realized by a supervisor that runs in parallel with the system and imposes the specification by disabling some of the controllable events in a feedback manner.

Decentralized supervisory control was developed by Rudie and Wonham [39]. It is based on the idea to distribute the actuator and sensor capabilities among several local supervisors. Each supervisor issues a control decision based on its own observation of the system. The global control action is then given by a fusion rule on the local control actions.

To give an example, consider a traffic system of a region with many elements (crossroads, tunnels), traffic lights and information boards. One of the goals may be to make the traffic fluent and prevent traffic jams in case of unexpected circumstances. The elements are observed (automatically or by humans) to provide information about the situation. One element may be observed by several supervisors corresponding to, e.g., different directions. The overall system is modeled as a single system including all dependences between the elements. The control decision is made based on all observations. Thus, although there are local supervisors that do not observe any problem, they need to react based on the observations of other supervisors, for instance, to close all the streets accessing a blocked tunnel.

There is an important motivation for decentralized supervisory control of DES that do not a priori have a modular structure. It is well known that the abstraction of timed automata into region (zone) automata does not preserve the modular structure. Similarly, it is to be expected that the discretization of a hybrid system does not preserve its modular structure. Then the original structure of a hybrid system is lost and we have to face the decentralized supervisory control problem instead of the modular supervisory control problem.

There are many different control policies based on two elementary ones: *conjunctive and permissive* (C & P) and *disjunctive and antipermissive* (D & A). For any decentralized control architecture, a corresponding notion of *coobservability* was proposed, which together with *controllability* form the necessary and sufficient conditions to achieve a specification by the controlled system. Nowadays, there are advanced architectures, such as an architecture with conditional decision (inferencing) [32, 55] or multi-level inferencing [23, 44]. A general approach consisting in several decentralized supervisory control architectures running in parallel was proposed by Chakib and Khoumsi [6].

Another approach to ensure coobservability is to extend locally observable events via communication among local supervisors. There exist decentralized control problems that cannot be solved without communication. Decentralized control with communicating supervisors, where an occurrence of transitions visible to one supervisor can be communicated to other supervisors, has been studied [1, 33, 37].

So far, almost all results available in the literature are only existential. There exist a few papers providing constructive results to compute a controllable and coobservable sublanguage of a specification, but the problem is in general computationally difficult. The existence of local supervisors enforcing the safety specification is decidable if nonblockingness is not required (e.g., for prefix-closed languages). However, if the marked language of the controlled system has to be included in the specification so that the controlled system is nonblocking, the existence of such local supervisors is undecidable [46, 47].

We focus on the computation of a controllable and coobservable sublanguage by using additional communications among the supervisors. Our study is restricted to the original (C & P) architecture [39, 54] and motivated by the relationship between decentralized and modular supervisory control and their key concepts: C & P coobservability and separability. This relationship is investigated in Komenda et al. [12], where the decentralized framework is plugged into the modular framework. The approach is based on the concurrent (separable) over-approximation of the plant. In decentralized control, there is no assumption on the structure of the plant. Therefore, both the system and the specification are replaced with their infimal separable superlanguages. However, in the likely case the specification fails to be separable, one only computes a solution for this new specification, which often fails to be included in the specification. Otherwise stated, it is assumed that the constructed sublanguage is included in the specification [12].

This is the point, where our approach comes into the picture. We overcome the problem of an inseparable specification by making it separable via communication (using the notion of *conditional decomposability* [15]). The approach of Komenda et al. [12] further requires *mutual controllability* to ensure coobservability of the separable over-approximation of the specification. This can be omitted in our approach, although we show that it is useful to ensure the centralized optimality.

Decomposable over-approximations were also considered by Jiang and Kumar [10], where decomposability of the specification is an additional assumption, whereas in our approach, it is enforced by the construction via additional communications.

In this paper, we construct a controllable and coobservable sublanguage of a specification with respect to possibly extended local observations (enriched by communication). The idea is different from the computation of purely coobservable sublanguages in the literature. First, we extend the set of locally observable events via communication of events observable by other supervisors. To compute such an extension, we use the technique for conditional decomposability [15]. Then we use the extended alphabets to compute the local supervisors in a fully decentralized way. We show that our construction guarantees coobservability (Theorem 7) as soon as the local supervisors are nonconflicting (in particular if they are prefix-closed). We then state two sufficient conditions under which the solution coincides with the centralized optimal solution. One approach uses mutual controllability (Theorem 10), which is a condition considering the structure of the plant, and the other the observer and LCC conditions (Theorem 11), which rather considers the structure of the supervisors. Since our approach is not restricted to prefix-closed languages, we further show how to handle the case of conflicting supervisors (Theorem 12). The complexity of our approach is briefly discussed in Section 5.5. We further show that separability is PSPACE-complete (Theorem 2), which generalizes a result on the complexity of decomposability.

Although our approach is different from the existing approaches, the rough idea is analogous to that used for decentralized synthesis without communication [21, 43]. The approach of Kozák and Wonham [21] is similar to ours in that it computes fully decentralized solutions that guarantee coobservability. It essentially projects the centralized supervisor, whereas we compute local supervisors and coobservability is guaranteed by construction (from separability that is granted by distributed computation). The main difference is that we do not project the centralized supervisor, but rather the plant and the specification. Also, the condition of Takai [43] under which the fully decentralized supervisors achieve the centralized optimal solution (the centralized optimal supervisor must be observable with respect to all locally observable alphabets) is different from our condition that relies on structural properties of the plant: such as observer and local control consistency or mutual controllability and observability of projections of the plant.

Our approach is also related to the topic on sensor selection [11, 34, 48, 49], where there is a maximal observable event set and the sensors can be turned on and off. In our approach, we have static observations, rather than dynamic, and communicate them among supervisors, where needed. Our work can be extended to dynamic observations in the future.

This work is an extended version of the conference paper [13]. We extend our approach to non-prefixed-closed languages and to partial observations, compare it to the centralized optimal solution, and include a discussion on the complexity of related problems.

2 Preliminaries

We assume that the reader is familiar with decentralized supervisory control of discrete-event systems [5, 52]. Let A be a finite nonempty set (an *alphabet*), and let A^* denote the set of all finite words over A; the empty word is denoted by ε. A *language* over A is a subset of A^*. The *prefix closure* of a language L over A is the set $\overline{L} = \{w \in A^* \mid \text{there exists } u \in A^* \text{ such that } wu \in L\}$. A language L is *prefix-closed* if $L = \overline{L}$.

A *generator* is a quintuple $G = (Q, A, f, q_0, Q_m)$, where Q is a finite set of *states*, A is an alphabet (of *events*), $f\colon Q \times A \to Q$ is a *partial transition function*, $q_0 \in Q$ is the *initial state*, and $Q_m \subseteq Q$ is a set of *marked states*. The transition function f can be extended to the domain $Q \times A^*$ in the usual way. The language *generated* by G is the set $L(G) = \{s \in$

$A^* \mid f(q_0,s) \in Q\}$ and the language *marked* by G is the set $L_m(G) = \{s \in A^* \mid f(q_0,s) \in Q_m\}$. The paper is restricted to *regular languages*, that is, languages marked by a generator.

A *projection* $P_o\colon A^* \to A_o^*$ is a morphism defined by $P_o(a) = \varepsilon$, if $a \in A \setminus A_o$, and $P_o(a) = a$, if $a \in A_o$. It is extended (as a morphism for concatenation) from events to words by induction. The *inverse image* of P_o is defined as $P_o^{-1}(a) = \{s \in A^* \mid P_o(s) = a\}$. These definitions can naturally be extended to languages. For two alphabets $A_x, A_y \subseteq A$, we use the notation P_y^x to denote a projection from A_x^* to A_y^* and we write simply P_y if $A_x = A$.

A *synchronous product* of languages $L_i \subseteq A_i^*$, for $i = 1,\ldots,n$, is defined as $\|_{i=1}^n L_i = \bigcap_{i=1}^n P_i^{-1}(L_i) \subseteq A^* = (\bigcup_{i=1}^n A_i)^*$, where $P_i\colon A^* \to A_i^*$ is a projection. Languages L_i are *synchronously nonconflicting* if $\overline{\|_{i=1}^n L_i} = \|_{i=1}^n \overline{L_i}$.

Let L be a prefix-closed language over an alphabet A. A language $K \subseteq L$ is *observable* with respect to L and projection $P_o\colon A^* \to A_o^*$ if, for all $s \in \overline{K}$ and $a \in A_c$, $sa \notin \overline{K}$ and $sa \in L$ implies that $P_o^{-1}P_o(s)\{a\} \cap \overline{K} = \emptyset$. Language K is *normal* with respect to L and projection P_o if $\overline{K} = P_o^{-1}P_o(\overline{K}) \cap L$.

2.1 Decentralized Supervisory Control

A *controlled generator* over an alphabet A is a structure $(G, (A_{c,i})_{i=1}^n, (A_{o,i})_{i=1}^n)$, where G is a generator over A, $A_{c,i} \subseteq A$ are sets of *locally controllable events*, and $A_{o,i} \subseteq A$ are sets of *locally observable events*. Let $A_c = \bigcup_{i=1}^n A_{c,i}$ denote the set of controllable events, $A_o = \bigcup_{i=1}^n A_{o,i}$ the set of observable events, $A_{uc} = A \setminus A_c$ the set of uncontrollable events, and $A_{uo} = A \setminus A_o$ the set of unobservable events. Projections to locally observable events $A_{o,i}$ are denoted by $P_{o,i}\colon A^* \to A_{o,i}^*$.

Let $\Gamma_i = \{\gamma \subseteq A \mid \gamma \supseteq (A \setminus A_{c,i})\}$ be a set of local control patterns. A supervisor S_i is a mapping $S_i\colon P_{o,i}(L(G)) \to \Gamma_i$, where $S_i(s)$ is the set of locally enabled events if S_i observes $s \in A_{o,i}^*$. The global control law S is the conjunction of local supervisors S_i given by $S(w) = \bigcap_{i=1}^n S_i(P_{o,i}(w))$ for $w \in A^*$. The *closed-loop system* is the smallest language $L(S/G)$ such that $\varepsilon \in L(S/G)$ and if $s \in L(S/G)$, $sa \in L(G)$ and $a \in S(s)$, then $sa \in L(S/G)$. Control objectives of decentralized control are defined using a specification language K. Let $L_m(S/G) = L(S/G) \cap K$. If $\overline{L_m(S/G)} = L(S/G)$, the closed-loop system is called *nonblocking*. The goal of decentralized control is to find supervisors $(S_i)_{i=1}^n$ such that $L_m(S/G) = K$ and $\overline{L_m(S/G)} = \overline{K}$.

Necessary and sufficient conditions to achieve a specification by a joint action of local supervisors are controllability and coobservability [39]. Let L be a prefix-closed language over A. A language $K \subseteq L$ is *controllable* with respect to L and the set of uncontrollable events A_{uc} if $\overline{K}A_{uc} \cap L \subseteq \overline{K}$. A language $K \subseteq L$ is *coobservable* with respect to L and the sets of locally observable events $(A_{o,i})_{i=1}^n$ if for all $s \in \overline{K}$, $a \in A_c$, and $sa \in L \setminus \overline{K}$, there exists $i \in \{1,2,\ldots,n\}$ such that $a \in A_{c,i}$ and $(P_{o,i}^{-1}(P_{o,i}(s))\{a\} \cap \overline{K} = \emptyset$. Intuitively, if, after a word s from the specification, the extension by an event a is illegal (it does not exist in the specification but exists in the plant), then there must exist at least one local supervisor S_i that can issue the decision "disable the event a".

The control law of local supervisors associated to the C & P architecture is called permissive, since the default action is to enable an event whenever a supervisor has an ambiguity what to do with it. Specifically, the control law of supervisor S_i on s is defined as $S_i(s) = (A \setminus A_{c,i}) \cup \{a \in A_{c,i} \mid \text{there exists } s' \in K \text{ with } P_{o,i}(s') = P_{o,i}(s) \text{ and } s'a \in K\}$. With the permissive local policy, we always achieve all words in the specification. The concern is then *safety*, expressed by coobservability.

Let $X \subseteq A$ be an alphabet. In the rest of the paper, we use the convention to define the set of uncontrollable events of X as $X_{uc} = X \cap A_{uc}$. Similarly, we define the set of controllable events of X as $X_c = X \cap A_c$, the set of observable events of X as $X_o = X \cap A_o$, and the set of unobservable events of X as $X_{uo} = X \cap A_{uo}$.

3 Main Idea of our Approach

We now present the main idea of our approach to compute a controllable and coobservable sublanguage of a specification language using communication and results of modular supervisory control.

Let $(G, (A_{c,i})_{i=1}^n, (A_{o,i})_{i=1}^n)$ be a controlled generator over an alphabet A. For simplicity, we denote $L = L(G)$. Let $K \subseteq L$ be a specification language over A. If the local supervisors do not observe all events of A, that is, A_{uo} is nonempty, we consider an arbitrary decomposition of A_{uo} into (not necessarily disjoint) local sets $A_{uo,i}$, such that the union of all $A_{uo,i}$ results in A_{uo}. The alphabet B_i of supervisor S_i contains all events from $A_{o,i} \cup A_{uo,i}$ and may be further extended with other events via communication. The union of all the alphabets B_i results in A. In Section 4, we suggest a procedure how to obtain such a decomposition of A_{uo}. Namely, for every alphabet $A_{o,i}$, we make use of the procedure RCD defined on page 6 below that computes an extension alphabet $\Sigma_i \subseteq A$. This extension can be decomposed into observable and unobservable events $\Sigma_{o,i}$ and $\Sigma_{uo,i}$ with respect to global observable and unobservable alphabets A_o and A_{uo}, respectively. The alphabet B_i of supervisor S_i is then the union of alphabets $A_{o,i}$, $\Sigma_{o,i}$, and $\Sigma_{uo,i}$. The events of the alphabet $A_{o,i} \cup \Sigma_{o,i}$ are the events observed by supervisor S_i extended with communications.

The idea of our approach is to compute local languages (supervisors) $\mathbf{R}_i$ over the alphabets B_i such that their synchronous product $\mathbf{R} = \|_{i=1}^n \mathbf{R}_i$ is a sublanguage of K controllable and coobservable with respect to L. Although there are well-known conditions on local languages in modular supervisory control that ensure that their synchronous product is controllable, cf. Lemma 14 in the appendix, conditions on local languages that ensure coobservability of their synchronous product are not known. We now identify two such sufficient conditions in Theorem 1.

Theorem 1. *Let L be a prefixed-closed language over $B = \bigcup_{i=1}^{n} B_i$ and assume that $B_{o,i} \cap B_c \subseteq B_{c,i}$, for $i = 1, \ldots, n$. Let $M \subseteq L$ be a language such that $\overline{M} = \|_{i=1}^{n} \overline{M_i}$, where M_i is a language over B_i. If*

1. *either M_i is normal with respect to $P_i(L)$ and $P_{o,i}^i$, for all $i = 1, \ldots, n$,*
2. *or $B_c \subseteq B_o$ and M_i is observable with respect to $P_i(L)$ and $P_{o,i}^i$, for all $i = 1, \ldots, n$,*

then M is coobservable with respect to L and $(B_{o,i})_{i=1}^n$.

Proof. For the sake of contradiction, assume that language $\overline{M}$ is not coobservable with respect to L and $(B_{o,i})_{i=1}^n$. Then there exist $s \in \overline{M}$ and $a \in B_c$ such that $sa \in L \setminus \overline{M}$ and, for each $i \in \{1, 2, \ldots, n\}$, either $a \notin B_{c,i}$ or $P_{o,i}^{-1} P_{o,i}(s)\{a\} \cap \overline{M} \neq \emptyset$. Let $t = P_i(s) \in P_i(\overline{M}) \subseteq \overline{M_i}$. Then $sa \in L$ implies that $tP_i(a) \in P_i(L)$. We show below that $sa \in P_i^{-1}(\overline{M_i})$. It then completes the proof, since $sa \in \bigcap_{i=1}^{n} P_i^{-1}(\overline{M_i}) = \overline{M}$, which is a contradiction with $sa \in L \setminus \overline{M}$.

If $a \in B_{c,i}$, then there exists s_i such that $s_i a \in \overline{M}$ and $P_{o,i}(s_i) = P_{o,i}(s)$. Let $t_i = P_i(s_i) \in P_i(\overline{M}) \subseteq \overline{M_i}$. Then $s_i a \in \overline{M}$ implies that $t_i a \in P_i(\overline{M}) \subseteq \overline{M_i}$. Moreover, $P_{o,i}^i(t) = P_{o,i}^i(P_i(s)) = P_{o,i}(s) = P_{o,i}(s_i) = P_{o,i}^i(P_i(s_i)) = P_{o,i}^i(t_i)$. Since $t_i a \in \overline{M_i}$ and $t_i a \in (P_{o,i}^i)^{-1} P_{o,i}^i(t)\{a\}$, observability of M_i with respect to $P_i(L)$ and $P_{o,i}^i$ implies that $ta = P_i(sa) \in \overline{M_i}$, that is, $sa \in P_i^{-1}(\overline{M_i})$.

If $a \notin B_{c,i}$, then $a \notin B_{o,i} \cap B_c$. Since $a \in B_c$, we have that $a \notin B_{o,i}$. If $a \notin B_{uo,i}$, then $a \notin B_i$. Hence $P_i(a) = \varepsilon$, and $P_i(sa) = P_i(s) \in P_i(\overline{M}) \subseteq \overline{M_i}$ implies that $sa \in P_i^{-1}(\overline{M_i})$. If $a \in B_{uo,i}$, then normality of M_i with respect to $P_i(L)$ implies that $\overline{M_i} = (P_{o,i}^i)^{-1} P_{o,i}^i(\overline{M_i}) \cap P_i(L)$. Since $t \in \overline{M_i}$ and $P_{o,i}^i(a) = \varepsilon$, we have that $ta \in (P_{o,i}^i)^{-1} P_{o,i}^i(\overline{M_i})$. Then $ta \in P_i(L)$ implies that $ta \in \overline{M_i}$, and $ta = P_i(sa)$ then gives that $sa \in P_i^{-1}(\overline{M_i})$. This completes the proof for M_i normal.

If M_i is not normal, we have that $B_c \subseteq B_o$ and M_i is observable. Then $a \in B_c$ implies that $a \notin B_{uo}$, hence the only possible case is $a \notin B_{uo,i}$, which we have already shown above. □

The way we compute the languages $\mathbf{R}_i$ is as follows. We decompose specification K in such a way that $K = \|_{i=1}^{n} K_i$, where K_i are languages over B_i, and over-approximate L by the synchronous product of its projections $P_i(L)$ on alphabets B_i. The condition required on K does not always hold and is equivalent to the notion of *separability* defined below. How to construct the alphabets B_i so that K satisfies the separability condition is discussed in Section 4. The languages $\mathbf{R}_i$ are then computed locally as sublanguages or superlanguages of K_i that satisfy the sufficient conditions (Lemma 14 and Theorem 1) that make their synchronous product $\mathbf{R}$ controllable, coobservable and included in K. These computations are discussed in Section 5.

In Theorem 1, we assume that if a supervisor observes a controllable event, it can also control it; that is, $B_{o,i} \cap B_c \subseteq B_{c,i}$. This is a new condition that deserves a discussion. Rudie and Wonham [39] showed that under the assumption that a supervisor can always observe the events it can control, that is, $B_{c,i} \subseteq B_{o,i}$, decomposability (a generalization of separability) is equivalent to coobservability. Our condition $B_{o,i} \cap B_c \subseteq B_{c,i}$ is weaker in the sense that it does not require that $B_{c,i}$ is included in $B_{o,i}$. Similarly, the assumption $B_c \subseteq B_o$ in case 2 of Theorem 1 does not mean that $B_{c,i}$ is included in $B_{o,i}$. It only requires that every controllable event is observed by one of the supervisors.

To justify our assumption, let a be a controllable event that is observable by a supervisor S_i. If a is not physically controllable by S_i, we can still make use of the advantage that S_i observes a. Namely, S_i may provide information about a as if a was controllable for it. The fusion rule (global supervisor S) then decides which events need to be disabled in the current situation, and communicates this decision back to the supervisors. If S_i requires that a needs to be disabled, the global supervisor will require to disable a. Since a is controllable, there must be a local supervisor S_j that can physically control a. Then supervisor S_j will take care of disabling a. Another view is that if S_i finds out that a needs to be disabled, it communicates this observation directly to S_j, which takes the corresponding actions.

4 Separability and Communication

In this section, we define the notion of separability and suggest a procedure to construct the alphabets B_i so that K is separable with respect to $(B_i)_{i=1}^n$ as required in our approach.

A conceptually simpler condition than coobservability is known in the literature as decomposability [38]. A language K over A is *decomposable* with respect to alphabets $(A_i)_{i=1}^n$ and L if $K = (\bigcap_{i=1}^{n} P_i^{-1} P_i(K)) \cap L$, where P_i is the projection from A^* to A_i^*. The inclusion $K \subseteq \bigcap_{i=1}^{n} P_i^{-1} P_i(K) \cap L$ holds true whenever $K \subseteq L$. A special case of decomposability for $L = A^*$ is known as *separability* [50]. For $A = \bigcup_{i=1}^{n} A_i$, we can replace intersection in the definition by parallel composition. Namely, language K is separable with respect to alphabets $(A_i)_{i=1}^n$ if $K = \|_{i=1}^{n} P_i(K)$. As already pointed out, separability of K with respect to alphabets $(A_i)_{i=1}^n$ is equivalent to the existence of languages K_i over A_i such that the synchronous product of K_i results in K. Then $P_i(K)$ is included in K_i, hence the languages $P_i(K)$ are the minimal (infimal) languages (with respect to inclusion) whose synchronous product results in K.

From the computational point of view, the first question is the complexity of deciding whether language K is separable with respect to alphabets $(A_i)_{i=1}^n$. We show that the problem is PSPACE-complete. A decision problem is PSPACE-complete if it can be solved in polynomial space with respect to the size of the input and if every problem that can be solved in polynomial space can be reduced to it in polynomial time. A proof of the following result can be found in the appendix.

Theorem 2. *The following problem is PSPACE-complete.*

INPUT: *Alphabets $A_1, A_2, \ldots, A_n$ and a generator H over $\bigcup_{i=1}^n A_i$.*

OUTPUT: `Yes` *if and only if $L_m(H)$ is separable with respect to $(A_i)_{i=1}^n$.*

The size of the input is the number of states and transitions of the generator H and the size of the n alphabets $A_1, \ldots, A_n$. The verification can be done in polynomial time by a direct computation if the number of alphabets is restricted by a constant. Therefore, the unrestricted number of alphabets (supervisors) is what makes the problem computationally difficult.

Separability is a special case of decomposability where L is universal. As a consequence, decomposability of K with respect to $(A_i)_{i=1}^n$ and L is PSPACE-complete even if $L = A^*$. This generalizes a result that can be derived from the literature. Namely, coobservability of K with respect to $(A_i)_{i=1}^n$ and L is known to be PSPACE-complete [36]. Under some assumptions [39, Proposition 4.3], decomposability is equivalent to coobservability. Since the reduction by Rohloff et al. [36] satisfies these assumptions, decomposability of K with respect to $(A_i)_{i=1}^n$ and L is PSPACE-complete. In those proofs, however, L is different from A^*, hence our theorem generalizes this result.

Another question, which we have to face, is what to do if language K is not separable with respect to $(A_i)_{i=1}^n$. In this case, it would be natural to take a maximal (with respect to inclusion) sublanguage of K that is separable with respect to $(A_i)_{i=1}^n$. Unfortunately, Lin et al. [27] have shown that to find such a maximal sublanguage is not algorithmically possible.

To overcome these issues – high complexity and undecidability – we use the notion of conditional decomposability [15]. A language K is *conditionally decomposable* with respect to alphabets $(A_i)_{i=1}^n$ and an alphabet Σ if Σ contains all shared events, that is, $\bigcup_{i \neq j}(A_i \cap A_j) \subseteq \Sigma \subseteq \bigcup_{i=1}^n A_i$, and K is separable with respect to alphabets $(A_i \cup \Sigma)_{i=1}^n$, that is, $K = \|_{i=1}^n P_{i+\Sigma}(K)$, where $P_{i+\Sigma}$ denotes the projection from A^* to $(A_i \cup \Sigma)^*$. Conditional decomposability thus requires to find an alphabet Σ containing all shared events such that K is separable with respect to $(A_i \cup \Sigma)_{i=1}^n$.

Compared to separability, there are two advantages of conditional decomposability. First, every language can be made conditionally decomposable by finding a convenient alphabet Σ. Such an alphabet always exists; indeed, one could take $\Sigma = \bigcup_{i=1}^n A_i$, but the aim is to find a reasonably small alphabet. This advantage of conditional decomposability helps us overcome the undecidable issue of finding a maximal nonempty separable sublanguage. The second advantage is a lower complexity of checking conditional decomposability. To check whether K is conditionally decomposable with respect to $(A_i)_{i=1}^n$ and Σ can be done in polynomial time, compared to PSPACE for separability. What allows this efficiency is the assumption that Σ contains all shared events. The following theorem is a generalization of a result obtained for pairwise disjoint alphabets by Willner and Heymann [50].

Theorem 3 ([15])**.** *Let K be a language represented as a generator, and let $(A_i)_{i=1}^n$ and Σ be alphabets. The problem to decide whether K is conditionally decomposable with respect to $(A_i)_{i=1}^n$ and Σ can be solved in polynomial time.*

Recall that our idea to compute the controllable and coobservable sublanguage involves an over-approximation of the plant language L by a new modular plant $\|_{i=1}^n P_{i+\Sigma}(L)$. We show in the following lemma that it is better to consider the projections $P_{i+\Sigma}(L)$ rather than $P_i(L)$ used in Komenda et al. [12] because the larger the extension Σ, the better the over-approximation of L.

Lemma 4. *Let $(A_i)_{i=1}^n$ be alphabets, and let L be a language over the alphabet $A = \bigcup_{i=1}^n A_i$. Let $\Sigma \subseteq A$ be an alphabet, and let $P_i \colon A^* \to A_i^*$ and $P_{i+\Sigma} \colon A^* \to (A_i \cup \Sigma)^*$ be projections. Then $L \subseteq \|_{i=1}^n P_{i+\Sigma}(L) \subseteq \|_{i=1}^n P_i(L)$.*

Proof. The first inclusion holds for any projection. To prove the other inclusion, notice that it holds that $P_{i+\Sigma}(L) \subseteq (P_i^{i+\Sigma})^{-1}(P_i^{i+\Sigma}(P_{i+\Sigma}(L))) = (P_i^{i+\Sigma})^{-1}(P_i(L))$, where $P_i^{i+\Sigma}$ is the projection P_i restricted to the domain $(A_i \cup \Sigma)^*$. Then $P_{i+\Sigma}^{-1}(P_{i+\Sigma}(L)) \subseteq P_{i+\Sigma}^{-1}(P_i^{i+\Sigma})^{-1}(P_i(L)) = P_i^{-1}P_i(L)$, which completes the proof. □

It may seem that the largest Σ is the best choice. However, a larger Σ means more communication or more sensors to observe the system (the local supervisors need to observe more). On the other hand, to compute a minimal extension Σ with respect to the cardinality is an NP-hard problem [17]. Nevertheless, there is an algorithm to find an acceptable extension in polynomial time [15]. In general, to find a suitable extension in a reasonable time is an interesting research topic. The choice of Σ can be influenced by several factors, such as the price of sensors, the (im)possibility to observe an event (by a specific local supervisor) etc.

Even if we consider the minimal extension Σ, it may happen that too many events are forced to be communicated between all supervisors though it is not needed. We demonstrate this in the following example. It shows that it is more convenient to search for some local alphabets so that the specification is separable with respect to them.

Example 1. Consider the language K over $A = \{a, b, c, d, e, f\}$, whose generator is depicted in Fig. 1, and the alphabets $A_{o,1} = \{a, e\}$, $A_{o,2} = \{b, e\}$, $A_{o,3} = \{c, f\}$, and $A_{o,4} = \{d, f\}$. To make K conditionally decomposable with respect to $(A_{o,i})_{i=1}^4$ and Σ, the extension Σ must contain at least all shared events, that is, e and f; actually, $\Sigma = \{e, f, a, c\}$ is a minimal extension making K conditionally decomposable with respect to $(A_{o,i})_{i=1}^4$ and Σ. However, the reader may notice that a needs to be communicated only between the supervisors S_1 and S_2, whereas c needs to be communicated only between the supervisors S_3 and S_4. Specifically, K is conditionally decomposable with respect to alphabets $A_{o,1} \cup A_{o,2}$, $A_{o,3} \cup A_{o,4}$ and

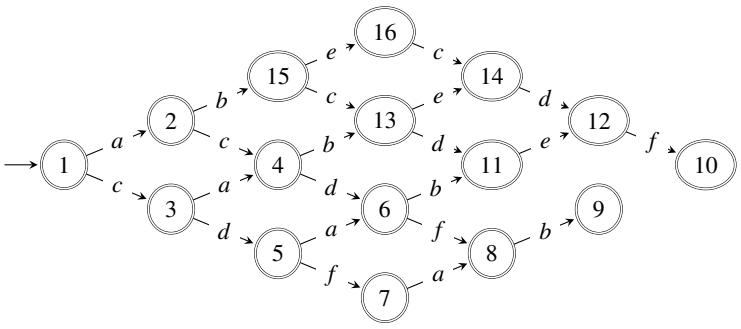

Figure 1: Generator for language K of Example 1

$\Sigma_{all} = \{f\}$; that is, $K = P_{\{a,b,e,f\}}(K) \parallel P_{\{c,d,f\}}(K)$. Having this, notice that language $P_{\{a,b,e,f\}}(K) = \overline{\{abef, afb, fab\}}$ is conditionally decomposable with respect to $A_{o,1} \cup \Sigma_{all}$, $A_{o,2} \cup \Sigma_{all}$ and $\Sigma_{1,2} = \{a,e,f\}$, and language $P_{\{c,d,f\}}(K) = \overline{\{cdf\}}$ is conditionally decomposable with respect to $A_{o,3} \cup \Sigma_{all}$, $A_{o,4} \cup \Sigma_{all}$ and $\Sigma_{3,4} = \{c,f\}$. This means that a and c are communicated only locally and only f is communicated globally. In other words, K is separable with respect to alphabets $\{a,e,f\}$, $\{a,b,e,f\}$, $\{c,f\}$, and $\{c,d,f\}$.

The reader can compare this with the purely conditional decomposable case computing the single extension $\Sigma = \{e,f,a,c\}$ making the language K separable with respect to $(A_{o,i} \cup \Sigma)_{i=1}^{n}$, that is, with respect to alphabets $\{a,c,e,f\}$, $\{a,b,c,e,f\}$, $\{a,c,e,f\}$, and $\{a,c,d,e,f\}$. □

In Example 1 we need to check separability of K with respect to alphabets $A_{o,1} \cup A_{o,2} \cup \Sigma_{all} = \{a,b,e,f\}$ and $A_{o,3} \cup A_{o,4} \cup \Sigma_{all} = \{c,d,f\}$. According to Theorem 3, this can be done in polynomial time. Moreover, the alphabets Σ_{all}, $\Sigma_{1,2}$ and $\Sigma_{3,4}$ are computed in polynomial time. This suggests the following refinement of the conditional decomposability procedure.

Procedure: REFINED CONDITIONAL DECOMPOSABILITY (RCD)

Let $(A_i)_{i=1}^{n}$ be alphabets such that $A_i \neq A_j$ for $i \neq j$, and let K be a language over $\bigcup_{i=1}^{n} A_i$. We define the equivalence relation $\sim$ as the minimal equivalence relation such that $A_i \sim A_j$ if A_i and A_j are not disjoint ($A_i \cap A_j \neq \emptyset$). The following steps of the procedure are illustrated in Example 2 below.

1. Let $A_i/_\sim = \{A_j \mid A_j \sim A_i, 1 \leq j \leq n\}$ be the equivalence class of alphabets equivalent to A_i. Let $C_i = \bigcup_{A_j \in A_i/_\sim} A_j$ be the set of all events that appear in $A_i/_\sim$. Let $\{C_{k_1}, \ldots, C_{k_m}\} = \{C_1, \ldots, C_n\}$, that is, we remove duplicates, hence we have $C_{k_i} \cap C_{k_j} = \emptyset$ for $k_i \neq k_j$.

2. We first compute a global extension Σ_{all} of events shared by the sets $(C_{k_i})_{i=1}^{m}$, which makes K conditionally decomposable with respect to $(C_{k_i})_{i=1}^{m}$ and Σ_{all}. This gives

$$K = \|_{i=1}^{m} P_{C_{k_i} \cup \Sigma_{all}}(K).$$

3. Then, for each local part $A_{k_i}/_\sim$, $i = 1,2,\ldots,m$, we make the language $P_{C_{k_i} \cup \Sigma_{all}}(K)$ conditionally decomposable with respect to $D_{k_i} = \{A_j \cup \Sigma_{all} \mid A_j \in A_{k_i}/_\sim\}$ and Σ_{k_i}. The set notation is used here to eliminate duplicates. If D_{k_i} is a singleton, we set $\Sigma_{k_i} = \Sigma_{all}$. This then gives that

$$P_{C_{k_i} \cup \Sigma_{all}}(K) = \|_{C \in D_{k_i}} P_{C \cup \Sigma_{k_i}}(K).$$

   Notice that if $C, C' \in D_{k_i}$, then $\Sigma_{all} \subseteq C \cap C'$, which implies that $\Sigma_{all} \subseteq \Sigma_{k_i}$.

4. Finally, for every $A_j \in A_{k_i}/_\sim$, we set $\Sigma_j = \Sigma_{k_i}$.

We now illustrate this procedure on a previous example.

Example 2. Consider Example 1. In this case, we have that $A_{o,1} \sim A_{o,2}$ and $A_{o,3} \sim A_{o,4}$. Therefore, $A_1/_\sim = A_2/_\sim = \{A_{o,1}, A_{o,2}\}$ and $A_3/_\sim = A_4/_\sim = \{A_{o,3}, A_{o,4}\}$. It further gives that $C_1 = C_2 = A_{o,1} \cup A_{o,2}$ and $C_3 = C_4 = A_{o,3} \cup A_{o,4}$. Then we compute $\Sigma_{all} = \{f\}$ such that K is conditionally decomposable with respect to C_1, C_3 and Σ_{all}. In other words, K is separable with respect to $C_1 \cup \Sigma_{all}$ and $C_3 \cup \Sigma_{all}$. The projections are $P_{C_1 \cup \Sigma_{all}} : A^* \to \{a,b,e,f\}^*$ and $P_{C_3 \cup \Sigma_{all}} : A^* \to \{c,d,f\}^*$. Then we compute $\Sigma_1 = \{a,e,f\}$ such that language $P_{C_1 \cup \Sigma_{all}}(K) = \overline{\{abef, afb, fab\}}$ is conditionally decomposable with respect to $A_{o,1} \cup \Sigma_{all}$, $A_{o,2} \cup \Sigma_{all}$ and Σ_1, and $\Sigma_3 = \{c,f\}$ such that language $P_{C_3 \cup \Sigma_{all}}(K) = \overline{\{cdf\}}$ is conditionally decomposable with respect to $A_{o,3} \cup \Sigma_{all}$, $A_{o,4} \cup \Sigma_{all}$ and Σ_3. Finally, we set $\Sigma_2 = \Sigma_1$ and $\Sigma_4 = \Sigma_3$. Language K is then separable with respect to $(A_{o,i} \cup \Sigma_i)_{i=1}^{4}$, that is, with respect to alphabets $B_1 = \{a,e,f\}$, $B_2 = \{a,b,e,f\}$, $B_3 = \{c,f\}$, and $B_4 = \{c,d,f\}$. □

We now show that the procedure is correct.

Theorem 5. *Let $(A_i)_{i=1}^n$ be alphabets such that $A_i \neq A_j$ for $i \neq j$, and let K be a language over $\bigcup_{i=1}^n A_i$. Let $(\Sigma_i)_{i=1}^n$ be the extensions computed by procedure* RCD. *Then K is separable with respect to $(A_i \cup \Sigma_i)_{i=1}^n$.*

Proof. Let $\{C_{k_1}, \ldots, C_{k_m}\} = \{C_1, \ldots, C_n\}$. Then $C_{k_i} \cap C_{k_j} = \emptyset$, for $k_i \neq k_j$, since C_{k_i} and C_{k_j} contain events of different classes. After the computation of Σ_{all}, $K = \|_{i=1}^m P_{C_{k_i} \cup \Sigma_{all}}(K)$. Then, for each D_{k_i}, $i = 1, \ldots, m$, we compute Σ_{k_i} such that $P_{C_{k_i} \cup \Sigma_{all}}(K) = \|_{C \in D_{k_i}} P_{C \cup \Sigma_{k_i}}(K) = \|_{A_j \in A_{k_i}/\sim} P_{A_j \cup \Sigma_{all} \cup \Sigma_j}(K)$, since $\Sigma_j = \Sigma_{k_i}$ and the synchronous product operation is idempotent. Together, we obtain that $K = \|_{i=1}^m \|_{A_j \in A_{k_i}/\sim} P_{A_j \cup \Sigma_{all} \cup \Sigma_j}(K) = \|_{i=1}^n P_{A_i \cup \Sigma_{all} \cup \Sigma_i}(K)$. Finally, since $\Sigma_{all} \subseteq \Sigma_i$, for all i, K is separable with respect to $(A_i \cup \Sigma_i)_{i=1}^n$. □

The procedure significantly depends on the computation and verification of conditional decomposability, which relies on the assumption that all shared events of the alphabets under (local) considerations are always included in Σ_{all} (resp. Σ_i). This then allows us to use Theorem 3 to check the property in polynomial time.

5 Computation of Coobservable Sublanguages

In this section, we discuss how to compute the languages $\mathbf{R}_i$ locally as sublanguages or superlanguages of K_i so that they satisfy the sufficient conditions that make their synchronous product controllable, coobservable and included in K.

Consider the settings of decentralized control, and let $(\Sigma_i)_{i=1}^n$ be extensions of local alphabets $(A_{o,i})_{i=1}^n$ computed by the procedure RCD described in Section 4, such that the specification K is separable with respect to $(A_{o,i} \cup \Sigma_i)_{i=1}^n$. We now apply results of modular control, where the status of an event is global. Namely, all shared events have the same status in all components where they appear. This is not in general the case in decentralized control. However, since every shared event appears in at least one Σ_i, the choice of $\Sigma_{uo,i}$ then ensures that the status of shared observable events is the same in all components where they appear. Recall that, for controllable events, we assume that if a supervisor observes a controllable event, then it can also control it. Formally, we assume that $A_{o,i} \cap A_c \subseteq A_{c,i}$, which must also hold after the extension, that is, $(A_{o,i} \cup \Sigma_{o,i}) \cap A_c \subseteq (A_{c,i} \cup \Sigma_{c,i})$, for $i = 1, 2, \ldots, n$. Therefore, we adapt the controllable status of events, if needed, to ensure this condition.

Before we proceed, we summarize our assumptions and notation as Assumption 6. It allows us to keep the rest of the paper more concise by referring to Assumption 6 rather than repeating the individual assumptions in the statements of theorems that follow.

Assumption 6. *Let $(G, (A_{c,i})_{i=1}^n, (A_{o,i})_{i=1}^n)$ be a controlled generator over an alphabet A. Let $L = L(G)$, and let $K \subseteq L$ be a specification language over A. Let $(\Sigma_i)_{i=1}^n$ be extensions of local alphabets $(A_{o,i})_{i=1}^n$ computed by the procedure* RCD, *such that language K is separable with respect to alphabets $(A_{o,i} \cup \Sigma_i)_{i=1}^n$, where the union of alphabets $A_{o,i} \cup \Sigma_i$ results in A, that is, $A = \bigcup_{i=1}^n (A_{o,i} \cup \Sigma_i)$.*

We further assume that if a supervisor observes a controllable event, then it can also control it; namely, that $(A_{o,i} \cup \Sigma_{o,i}) \cap A_c \subseteq (A_{c,i} \cup \Sigma_{c,i})$, where $\Sigma_{o,i} = \Sigma_i \cap A_o$ are the controllable events of Σ and $\Sigma_{uo,i} = \Sigma_i \setminus \Sigma_{o,i}$ are the uncontrollable events of Σ.

For $i = 1, 2, \ldots, n$, let $P_{i+\Sigma_i}$ denote the projection from A^* to $(A_{o,i} \cup \Sigma_i)^*$. Let $\mathbf{R}_i$ be languages that are controllable with respect to projection $P_{i+\Sigma_i}(L)$ of the plant language L to alphabet $A_{o,i} \cup \Sigma_i$ and locally uncontrollable events $(A_{o,i} \cup \Sigma_i)_{uc} = (A_{o,i} \cup \Sigma_i) \cap A_{uc}$ such that their synchronous product $\mathbf{R} = \|_{i=1}^n \mathbf{R}_i$ is included in K. Furthermore, we assume that $\mathbf{R}_i$ are

1. *either normal with respect to $P_{i+\Sigma_i}(L)$ and $A_{o,i} \cup \Sigma_{o,i}$,*
2. *or observable with respect to $P_{i+\Sigma_i}(L)$ and $A_{o,i} \cup \Sigma_{o,i}$, and all controllable events are observable ($A_c \subseteq A_o$).* □

Notice that normality implies observability [5], hence every local language $\mathbf{R}_i$ is observable. However, if one of the local languages $\mathbf{R}_i$ is observable and not normal, then we require that all controllable events are observable.

5.1 Main Result

We now state our main result showing how to use our framework to compute a controllable and coobservable sublanguage, and illustrate it on an example.

An important feature of our computation is that we automatically obtain a coobservable sublanguage.

Theorem 7. *Consider Assumption 6. If the languages $\mathbf{R}_i$ are synchronously nonconflicting (in particular, if they are prefix-closed), then $\mathbf{R} = \|_{i=1}^n \mathbf{R}_i$ is a sublanguage of K controllable with respect to L and A_{uc}, and coobservable with respect to L and $(A_{o,i} \cup \Sigma_{o,i})_{i=1}^n$.*

Proof. By definition, we have that $\mathbf{R} = \|_{i=1}^n \mathbf{R}_i \subseteq K$. By Lemmas 14 and 15 (Lemma 16) in the appendix, $\mathbf{R}$ is controllable and normal (observable) with respect to $\|_{i=1}^n P_{i+\Sigma_i}(L)$. Since $L \subseteq \|_{i=1}^n P_{i+\Sigma_i}(L)$, it is also controllable and normal (observable) with respect to L. Because $(A_{o,i} \cup \Sigma_{o,i}) \cap A_c \subseteq (A_{c,i} \cup \Sigma_{c,i})$ by the assumption and $\mathbf{R}_i$ are synchronously nonconflicting, that is, $\overline{\mathbf{R}} = \|_{i=1}^n \overline{\mathbf{R}_i}$, Theorem 1 implies that $\mathbf{R} = \|_{i=1}^n \mathbf{R}_i$ is coobservable with respect to L and $(A_{o,i} \cup \Sigma_{o,i})_{i=1}^n$. □

We now illustrate our approach on a simple example.

Example 3. Consider the languages $K = \overline{\{aa, ba, bbd, abc\}}$ and $L = \overline{\{aac, abc, bac, bbd\}}$ over $A = \{a,b,c,d\}$, and alphabets $A_{o,1} = A_{c,1} = \{a,c\}$ and $A_{o,2} = A_{c,2} = \{b,d\}$. Then K is not coobservable with respect to L and $(A_{o,i})_{i=1}^2$, because none of the supervisors is able to distinguish between *ab* and *ba*, where the continuation of *ba* by *c* within the plant leads outside the specification while the continuation of *ab* by *c* remains within the specification.

We compute the extensions $\Sigma_1 = \Sigma_2 \supseteq A_{o,1} \cap A_{o,2}$ by the procedure RCD such that K is separable with respect to $A_{o,1} \cup \Sigma_1$ and $A_{o,2} \cup \Sigma_2$. It is sufficient to take $\Sigma_1 = \Sigma_2 = \{b\}$, which needs to be communicated/observed by both supervisors. Since our system is with complete observations, we may compute $\mathbf{R}_1 = \overline{\{aa, abc, ba, bb\}}$ and $\mathbf{R}_2 = \overline{\{bbd\}}$ as the supremal controllable sublanguages of $P_{i+\Sigma_i}(K)$ with respect to $P_{i+\Sigma_i}(L)$ and $(A_{o,i} \cup \Sigma_i)_{uc}$. By Theorem 7, we have that $\mathbf{R}_1 \parallel \mathbf{R}_2$ is coobservable with respect to L and the extended alphabets $\{a,b,c\}$ and $\{b,d\}$. Indeed, supervisor S_1 that exerts the control power over the event *c* is now able to distinguish between the words *ab*, after which *c* should be allowed, and *ba*, after which *c* should be disabled. □

5.2 Construction of the Languages $\mathbf{R}_i$

There are many ways how to compute the languages $\mathbf{R}_i$ discussed in the literature. In the case of full local observations, it is natural to define the language

$$\mathbf{R}_i = \sup \mathrm{C}_{i+\Sigma_i} = \sup \mathrm{C}(P_{i+\Sigma_i}(K), P_{i+\Sigma_i}(L), (A_{o,i} \cup \Sigma_i)_{uc}) \tag{1}$$

as the supremal controllable sublanguage of $P_{i+\Sigma_i}(K)$ with respect to $P_{i+\Sigma_i}(L)$ and uncontrollable events $(A_{o,i} \cup \Sigma_i)_{uc}$. In the case of partial observations, we may define the language

$$\mathbf{R}_i = \sup \mathrm{CN}(P_{i+\Sigma_i}(K), P_{i+\Sigma_i}(L), (A_{o,i} \cup \Sigma_i)_{uc}, A_{o,i} \cup \Sigma_{o,i})$$

as the supremal controllable and normal sublanguage of $P_{i+\Sigma_i}(K)$ with respect to $P_{i+\Sigma_i}(L)$, $(A_{o,i} \cup \Sigma_i)_{uc}$ and $A_{o,i} \cup \Sigma_{o,i}$ [5, 2]. Similarly, if $A_c \subseteq A_o$, we can define $\mathbf{R}_i$ as the supremal controllable and relatively observable sublanguage [4, 18], or we can use any of the methods to compute a controllable and observable sublanguage discussed in the literature [7, 45, 53]. In these cases, we have that $\mathbf{R}_i \subseteq P_{i+\Sigma_i}(K)$, and separability of K then implies that the synchronous product $\|_{i=1}^n \mathbf{R}_i$ is included in K as required in Assumption 6.

However, we do not restrict language $\mathbf{R}_i$ to be included in $P_{i+\Sigma_i}(K)$. This allows us to define $\mathbf{R}_i$ in many different ways. For instance, we can define $\mathbf{R}_i$ as the infimal controllable (and normal/observable) superlanguage of $P_{i+\Sigma_i}(K)$ with respect to $P_{i+\Sigma_i}(L)$ and $(A_{o,i} \cup \Sigma_i)_{uc}$ (and $A_{o,i} \cup \Sigma_{o,i}$) as discussed in the literature [5, 24, 25, 40]. We can further combine the approaches so that one of the $\mathbf{R}_i$ can be computed as a sublanguage and another one as a superlanguage, etc. In such a case, we do not get the assumption $\|_{i=1}^n \mathbf{R}_i \subseteq K$ by construction, but we need to check it. It is in general a PSPACE-complete problem.[1] On the other hand, the advantage it brings is a potentially better (larger) solution as illustrated in Example 4.

Example 4. Let $L = \overline{\{ab, ba, bdau, dbau\}}$ be a language over $A = \{a,b,d,u\}$, and consider the alphabets $A_{o,1} = \{a,u\}$, $A_{c,1} = \{a,d\}$, $A_{o,2} = \{b,u\}$, $A_{c,2} = \{b\}$. Let $K = \overline{\{ab, ba, bd, db\}}$ be a specification. Then K is not coobservable with respect to L and $(A_{o,i})_{i=1}^2$ because, for $db \in K$, we have $a \in A_c$, $dba \in L \setminus K$, and $P_1(db)a = a \in K$ and $P_2(db)a = ba \in K$. Thus, none of the supervisors can disable *a* after the word *db*.

First, we compute only sublanguages. Let $\Sigma_1 = \Sigma_2 = \{d,u\}$ be the extensions of local observations computed by RCD. Then K is separable with respect to $(A_{o,i} \cup \Sigma_i)_{i=1}^2$. Notice that $P_{1+\Sigma_1}(K) = \{a,d,\varepsilon\}$ and $P_{2+\Sigma_2}(K) = \overline{\{bd,db\}}$, and that $P_{1+\Sigma_1}(L) = \overline{\{a,dau\}}$ and $P_{2+\Sigma_2}(L) = \overline{\{bdu,dbu\}}$. Then $\mathbf{R}_1 = \sup \mathrm{C}_{1+\Sigma_1} = P_{1+\Sigma_1}(K)$ and $\mathbf{R}_2 = \sup \mathrm{C}_{2+\Sigma_2} = \{b,d,\varepsilon\}$, and the solution $\mathbf{R}_1 \parallel \mathbf{R}_2 \subsetneq K$ gives us a strict subset of K.

On the other hand, we can obtain the whole K if we consider infimal controllable superlanguages of $P_{i+\Sigma_i}(K)$ instead of supremal controllable sublanguages. Then we obtain the infimal controllable superlanguages $\mathbf{R}_1 = P_{1+\Sigma}(K) = \sup \mathrm{C}_{1+\Sigma_1}$ and $\mathbf{R}_2 = P_{2+\Sigma}(K)$, and the solution $\mathbf{R}_1 \parallel \mathbf{R}_2 = K$ then gives us the whole specification as the resulting language. □

Another problem with infimal controllable superlanguages is that they do not exist for general languages, but only for prefix-closed languages. This issue can be avoided by the following choice of $\mathbf{R}_i$ based on the computation of prefix-closed superlanguages.

Lemma 8. *Consider Assumption 6. Let T_i be the prefix-closed infimal controllable (and normal/observable) superlanguage of $P_{i+\Sigma_i}(K)$ with respect to $P_{i+\Sigma_i}(L)$ and $(A_{o,i} \cup \Sigma_i)_{uc}$ (and $A_{o,i} \cup \Sigma_{o,i}$). Then $\mathbf{R}_i = P_{i+\Sigma_i}(K) \cup [T_i \setminus \overline{P_{i+\Sigma_i}(K)}]$ is controllable (and normal/observable) with respect to $P_{i+\Sigma_i}(L)$ and $(A_{o,i} \cup \Sigma_i)_{uc}$ (and $A_{o,i} \cup \Sigma_{o,i}$).*

Proof. We show that $\overline{\mathbf{R}_i} = T_i$. Since $P_{i+\Sigma_i}(K) \subseteq T_i$ and $\mathbf{R}_i \subseteq T_i$, we have that $\overline{\mathbf{R}_i} \subseteq T_i$. To show that $T_i \subseteq \overline{\mathbf{R}_i}$, let $w \in T_i$. If $w \notin \overline{P_{i+\Sigma_i}(K)}$, then $w \in \mathbf{R}_i$ by definition. If $w \in \overline{P_{i+\Sigma_i}(K)}$, then there exists v such that $wv \in P_{i+\Sigma_i}(K) \subseteq \mathbf{R}_i$, hence $w \in \overline{\mathbf{R}_i}$. □

[1] There is a simple reduction from the finite-state automata intersection problem: Given a set of deterministic finite automata $\{G_i\}_{i=1}^n$ over a common alphabet B, is $\bigcap_{i=1}^n L_m(G_i) = \emptyset$? The problem is PSPACE-complete [22]. Let $A \cap B = \emptyset$. It is not hard to see that $\bigcap_{i=1}^n L_m(G_i) = \emptyset$ if and only if $\bigcap_{i=1}^n (L_m(G_i) \cup K) \subseteq K$. The reduction is polynomial since every $L_m(G_i) \cup K$ is represented by a generator computed from G_i and the generator for K in polynomial time by the standard product construction.

5.3 Conditions for Optimality for Full Observations

In this subsection, we discuss conditions under which the solution of Theorem 7 is optimal in the sense of maximal permissiveness. That is, under which conditions the solution coincides with the supremal centralized supervisor, if it exists. Since no centralized optimal solution exists in the case of partial observations, we restrict our attention in this subsection only to the case of full observations, that is, we assume in this section that $A_{uo} = \emptyset$. However, we point out that a similar result to Proposition 9 can be obtain using the notions of *mutual normality* [20, Theorem 4.23] or *mutual observability* [19, Theorem 5.1] in the case the set A_{uo} is nonempty. Theorem 10 below can then be modified in the corresponding way.

For $i = 1,\dots,n$, let a language L_i over A_i be prefix-closed. Languages $(L_i)_{i=1}^n$ are *mutually controllable* if $L_j(A_{i,u} \cap A_j) \cap P_j P_i^{-1}(L_i) \subseteq L_j$, for $i, j = 1,2,\dots,n$ [26]. The following compatibility between supremal controllable sublanguages and the synchronous composition operator is known [26]. We state the result only for prefix-closed languages and refer the reader to Lee and Wong [26] for the conditions on general languages.

Proposition 9. *Assume that $A_{o,i} \cap A_c \subseteq A_{c,i}$. If the prefix-closed languages $L_i \subseteq A_i^*$ are mutually controllable, then $\|_{i=1}^n \sup\mathrm{C}(P_i(K), P_i(L), A_{i,u}) = \sup\mathrm{C}(\|_{i=1}^n P_i(K), \|_{i=1}^n P_i(L), A_{uc})$ holds true for any prefix-closed language $K \subseteq L$.*

It may be that $P_{i+\Sigma_i}(L)$ and $P_{j+\Sigma_j}(L)$ are mutually controllable for fairly small alphabets Σ_i and Σ_j. However, if we do not require that K is separable, we cannot guarantee that the resulting supremal controllable sublanguage is included in K. This is the main issue with the approach in Komenda et al. [12]. Therefore, in this paper, we compute the communications $(\Sigma_i)_{i=1}^n$ using the procedure RCD, such that K is separable with respect to $(A_{o,i} \cup \Sigma_i)_{i=1}^n$, to ensure the inclusion of the resulting supremal controllable sublanguage in K. In addition, if L is also separable with respect to $(A_{o,i} \cup \Sigma_i)_{i=1}^n$, which can again be ensured in the same way as for K, we obtain the optimal centralized solution.

Theorem 10. *Consider Assumption 6. Let $K \subseteq L$ be prefix-closed languages, and let K and L be separable with respect to $(A_{o,i} \cup \Sigma_i)_{i=1}^n$. Let $\sup\mathrm{C}_{i+\Sigma_i}$ be defined by Equation (1) above. If $P_{i+\Sigma_i}(L)$ and $P_{j+\Sigma_j}(L)$ are mutually controllable, for $i, j = 1,2,\dots,n$, then $\|_{i=1}^n \sup\mathrm{C}_{i+\Sigma_i} = \sup\mathrm{C}(K, L, A_{uc})$ is coobservable with respect to L and $(A_{o,i} \cup \Sigma_i)_{i=1}^n$.*

Proof. Since $\sup\mathrm{C}_{i+\Sigma_i} \subseteq P_{i+\Sigma_i}(K)$, $\|_{i=1}^n \sup\mathrm{C}_{i+\Sigma_i} \subseteq \|_{i=1}^n P_{i+\Sigma_i}(K) = K$ by separability of K. By Theorem 7, $\|_{i=1}^n \sup\mathrm{C}_{i+\Sigma_i}$ is controllable with respect to $\|_{i=1}^n P_{i+\Sigma_i}(L) = L$ and A_{uc}, hence coobservable with respect to L and $(A_{o,i} \cup \Sigma_i)_{i=1}^n$ by Theorem 1. Finally, by Proposition 9, $\|_{i=1}^n \sup\mathrm{C}_{i+\Sigma_i} = \sup\mathrm{C}(K, L, A_{uc})$. □

Whether mutual controllability can be fulfilled or not depends on the system. However, mutual controllability holds if all shared events are controllable. For instance, in Example 3, the only shared event between $A_{o,1} \cup \Sigma_1$ and $A_{o,2} \cup \Sigma_2$ is event b, which is controllable. Therefore, the languages $P_{1+\Sigma_1}(L)$ and $P_{2+\Sigma_2}(L)$ are mutually controllable, and Theorem 10 implies that the parallel composition $\mathbf{R}_1 \parallel \mathbf{R}_2$ coincides with the optimal monolithic solution $\sup\mathrm{C}(K, L, A_{uc})$, and is coobservable with respect to L and the extended alphabets $\{a, b, c\}$ and $\{b, d\}$.

Depending on the system, mutual controllability may be a strong condition. We now present a result that ensures optimality and is based on the notions of an L-observer and local control consistency (LCC).

A projection $P_k : A^* \to A_k^*$, for $A_k \subseteq A$, is an *L-observer* for a language $L \subseteq A^*$ if for all $s \in \overline{L}$, if $P_k(s)t \in P_k(L)$, then there exists $u \in A^*$ such that $su \in L$ and $P_k(u) = t$ [51, 3]. The co-domain of a projection can always be extended to fulfill the condition. Although to compute the minimal extension is NP-hard, there is a polynomial-time algorithm to find an acceptable extension [9, 30]. The property also prevents the state explosion when computing projections. If P is an L-observer, then the generator for $P(L)$ is not larger (usually much smaller) than the one for L.

Let L be a prefix-closed language over A, and let $A_k \subseteq A$. Projection $P_k : A^* \to A_k^*$ is *locally control consistent* (LCC) with respect to a word $s \in L$ if for all events $a_u \in A_k \cap A_{uc}$ such that $P_k(s)a_u \in P_k(L)$, it holds that either there does not exist any word $u \in (A \setminus A_k)^*$ such that $sua_u \in L$, or there exists a word $u \in (A_{uc} \setminus A_k)^*$ such that $sua_u \in L$. Projection P_k is LCC with respect to L if P_k is LCC for all words of L [42, 41]. Notice that LCC is a weaker condition than OCC defined in Zhong and Wonham [56].

Theorem 11. *Consider Assumption 6. If every projection $P_{i+\Sigma_i}$ is an L-observer and LCC for L, and the languages $\sup\mathrm{C}_{i+\Sigma_i}$ that are defined in (1) are synchronously nonconflicting (e.g., prefix-closed), then the language $\|_{i=1}^n \sup\mathrm{C}_{i+\Sigma_i} = \sup\mathrm{C}(K, L, A_{uc})$ is coobservable with respect to L and $(A_{o,i} \cup \Sigma_{o,i})_{i=1}^n$.*

Proof. The identity $\|_{i=1}^n \sup\mathrm{C}_{i+\Sigma_i} = \sup\mathrm{C}(K, L, A_{uc})$ has been shown in the literature [8, 42]. Since $A_{uo} = \emptyset$, Theorem 1 finishes the proof. □

Note that both properties L-observer and LCC can be ensured by further extending the alphabets Σ_i in polynomial time.

A similar result to Theorem 11 can be obtained for supremal controllable and normal sublanguages. A condition under which the parallel composition of local supremal controllable and normal languages equals to the global supremal controllable and normal sublanguage can be found in [14, Theorem 25].

5.4 Conflicting Supervisors

So far, the theorems require that the local languages $\mathbf{R}_i$ are synchronously nonconflicting. The remaining question is thus the case of conflicting local supervisors. It is in general a PSPACE-complete problem to decide whether a parallel composition (of an unspecified number) of generators is nonblocking [35]. However, Malik [28] shows that current computers can explore more than 100 million of states using explicit algorithms without any optimization techniques. Moreover, it is possible to use an L-observer to alleviate the computational effort as used in the following construction.

Theorem 12. *Consider Assumption 6. Let $L_C \subseteq \|_{i=1}^n P_{\Sigma'}(\mathbf{R}_i)$ be a language that is controllable and normal (observable) with respect to $P_{\Sigma'}(L)$, Σ'_{uc} and Σ'_o, where $\Sigma' \subseteq A$ contains all events shared by any pair of $\mathbf{R}_i$ and $\mathbf{R}_j$, for $i \neq j$, and $P_{\Sigma'} : A^* \to \Sigma'^*$ is an $\mathbf{R}_i$-observer, for $i = 1,\ldots,n$. If $(A_{o,i} \cup \Sigma_{o,i} \cup \Sigma'_o) \cap A_c \subseteq (A_{c,i} \cup \Sigma_{c,i} \cup \Sigma'_c)$, then $\|_{i=1}^n (\mathbf{R}_i \parallel L_C)$ is a sublanguage of K controllable (and normal/observable) with respect to L and A_{uc} (and A_o) that is coobservable with respect to $(A_{o,i} \cup \Sigma_{o,i} \cup \Sigma'_o)_{i=1}^n$, and whose components are synchronously nonconflicting.*

Proof. By definition and Lemma 18, we have that $L_C \subseteq \|_{i=1}^n P_{\Sigma'}(\mathbf{R}_i) = P_{\Sigma'}(\|_{i=1}^n \mathbf{R}_i) \subseteq P_{\Sigma'}(K)$. Thus, $\|_{i=1}^n (\mathbf{R}_i \parallel L_C) = (\|_{i=1}^n \mathbf{R}_i) \parallel L_C \subseteq K \parallel P_{\Sigma'}(K) = K$.

To prove nonconflictness, we make use of Lemma 17 in the appendix, which states that $\overline{\mathbf{R}_i \parallel L_C} = \overline{\mathbf{R}_i} \parallel \overline{L_C}$ if and only if $\overline{P_{\Sigma'}(\mathbf{R}_i) \parallel L_C} = \overline{P_{\Sigma'}(\mathbf{R}_i) \parallel \overline{L_C}}$. The second equation holds, because both sides are equal to $\overline{L_C}$. Using Lemma 17 again, we have that $\overline{\|_{i=1}^n (\mathbf{R}_i \parallel L_C)} = \|_{i=1}^n \overline{\mathbf{R}_i} \parallel \overline{L_C}$ if and only if $\overline{\|_{i=1}^n P_{\Sigma'}(\mathbf{R}_i \parallel L_C)} = \|_{i=1}^n \overline{P_{\Sigma'}(\mathbf{R}_i \parallel L_C)}$. Lemma 17 can be applied again, since $P_{\Sigma'}$ is an $(\mathbf{R}_i \parallel L_C)$-observer. It follows from Theorem 2 in Pena et al. [29] saying that a composition of observers is an observer – note that $P_{\Sigma'}$ is an $\mathbf{R}_i$-observer by assumption and an L_C-observer since it is an identity. Again, the latter equation holds, because $P_{\Sigma'}(\mathbf{R}_i \parallel L_C) = P_{\Sigma'}(\mathbf{R}_i) \parallel L_C = L_C$, by Lemma 18 and the definition of L_C. Thus, both sides are equal to $\overline{L_C}$. To summarize, $\overline{\|_{i=1}^n (\mathbf{R}_i \parallel L_C)} = \|_{i=1}^n (\overline{\mathbf{R}_i \parallel L_C}) = \|_{i=1}^n \overline{\mathbf{R}_i} \parallel \overline{L_C}$.

To prove controllability (and normality/observability), note that $\mathbf{R}_i$ is controllable (and normal/observable) with respect to $P_{i+\Sigma_i}(L)$, and L_C is controllable (and normal/observable) with respect to $P_{\Sigma'}(L)$. By Lemma 14 (Lemmas 15 and 16) in the appendix and the nonconflictness shown above, $\mathbf{R}_i \parallel L_C$ is controllable (and normal/observable) with respect to $P_{i+\Sigma_i}(L) \parallel P_{\Sigma'}(L)$ and $(A_{o,i} \cup \Sigma_i \cup \Sigma') \cap A_{uc}$. Similarly, $\|_{i=1}^n (\mathbf{R}_i \parallel L_C)$ is controllable (and normal/observable) with respect to $\|_{i=1}^n P_{i+\Sigma_i}(L) \parallel P_{\Sigma'}(L)$ and A_{uc}. Since $L \subseteq \|_{i=1}^n P_{i+\Sigma_i}(L)$ and $L \subseteq P_{\Sigma'}^{-1} P_{\Sigma'}(L)$, we have that $\|_{i=1}^n (\mathbf{R}_i \parallel L_C)$ is controllable (and normal/observable) with respect to L and A_{uc}.

Since $\mathbf{R}_i \parallel L_C$ is normal (observable) with respect to $P_{i+\Sigma_i}(L) \parallel P_{\Sigma'}(L)$, and it holds that $P_{i+\Sigma_i+\Sigma'}(L) \subseteq P_{i+\Sigma_i}(L) \parallel P_{\Sigma'}(L)$, we have that $\mathbf{R}_i \parallel L_C$ is normal (observable) with respect to $P_{i+\Sigma_i+\Sigma'}(L)$. Thus, using Theorem 1, $\|_{i=1}^n (\mathbf{R}_i \parallel L_C)$ is coobservable with respect to L and $(A_{o,i} \cup \Sigma_{o,i} \cup \Sigma'_o)_{i=1}^n$. In general, to apply Theorem 1, we need to adjust the controllable status of events, if needed, to satisfy $(A_{o,i} \cup \Sigma_{o,i} \cup \Sigma'_o) \cap A_c \subseteq (A_{c,i} \cup \Sigma_{c,i} \cup \Sigma'_c)$. □

In case of full observations, we strengthen the previous result by computing the language L_C as a sublanguage of $\|_{i=1}^n P_{\Sigma'}(\mathbf{R}_i)$ controllable with respect to $\|_{i=1}^n \overline{P_{\Sigma'}(\mathbf{R}_i)}$ rather than to $P_{\Sigma'}(L)$, which may result in a larger language L_C because $\|_{i=1}^n \overline{P_{\Sigma'}(\mathbf{R}_i)} \subseteq P_{\Sigma'}(L)$.

Theorem 13. *Consider Assumption 6. Let L_C be the supremal sublanguage of $\|_{i=1}^n P_{\Sigma'}(\mathbf{R}_i)$ that is controllable with respect to $\|_{i=1}^n \overline{P_{\Sigma'}(\mathbf{R}_i)}$ and Σ'_{uc}, where $\Sigma' \subseteq A$ contains all events shared by any pair of $\mathbf{R}_i$ and $\mathbf{R}_j$, for $i \neq j$, and $P_{\Sigma'} : A^* \to \Sigma'^*$ is an $\mathbf{R}_i$-observer, for $i = 1,\ldots,n$. If every projection $P_{i+\Sigma_i}$ is an L-observer and LCC for L, and the projection $P_{\Sigma'}$ is LCC for $\|_{i=1}^n \overline{\mathbf{R}_i}$, then $\|_{i=1}^n (\mathbf{R}_i \parallel L_C) = \sup \mathrm{C}(K, L, A_{uc})$ is coobservable with respect to L and $(A_{o,i} \cup \Sigma_i \cup \Sigma')_{i=1}^n$, whose components are synchronously nonconflicting. It is again under the assumption that $(A_{o,i} \cup \Sigma_i \cup \Sigma') \cap A_c \subseteq (A_{c,i} \cup \Sigma_{c,i} \cup \Sigma'_c)$.*

Proof. Let $\sup \mathrm{C} = \sup \mathrm{C}(K, L, A_{uc})$. We first show that $\|_{i=1}^n (\mathbf{R}_i \parallel L_C)$ is a subset of $\sup \mathrm{C}$ coobservable with respect to L and $(A_{o,i} \cup \Sigma_i \cup \Sigma')_{i=1}^n$, and its components are synchronously nonconflicting. Similarly as in the proof of Theorem 12, we can show that $\|_{i=1}^n (\mathbf{R}_i \parallel L_C) \subseteq K$ and that $\overline{\|_{i=1}^n (\mathbf{R}_i \parallel L_C)} = \|_{i=1}^n (\overline{\mathbf{R}_i \parallel L_C}) = \|_{i=1}^n \overline{\mathbf{R}_i} \parallel \overline{L_C}$.

To prove controllability of $\|_{i=1}^n (\mathbf{R}_i \parallel L_C)$ with respect to L and A_{uc}, note that $\mathbf{R}_i$ is controllable with respect to $\overline{\mathbf{R}_i}$ and $(A_{o,i} \cup \Sigma_i)_{uc}$, and L_C is controllable with respect to $\|_{j=1}^n \overline{P_{\Sigma'}(\mathbf{R}_j)}$ and Σ'_{uc}. By Lemma 14 in the appendix and the nonconflictness of $\mathbf{R}_i$ and L_C shown above, $\mathbf{R}_i \parallel L_C$ is controllable with respect to $\overline{\mathbf{R}_i} \parallel (\|_{j=1}^n \overline{P_{\Sigma'}(\mathbf{R}_j)})$ and $(A_{o,i} \cup \Sigma_i \cup \Sigma')_{uc}$. Using the same argument on $\mathbf{R}_i \parallel L_C$, for $i = 1,\ldots,n$, we obtain that $\|_{i=1}^n (\mathbf{R}_i \parallel L_C)$ is controllable with respect to $\|_{i=1}^n (\overline{\mathbf{R}_i} \parallel \|_{j=1}^n \overline{P_{\Sigma'}(\mathbf{R}_j)}) = \|_{i=1}^n \overline{\mathbf{R}_i}$ and A_{uc}. One more application of Lemma 14 on $\overline{\mathbf{R}_i}$ gives that $\|_{i=1}^n \overline{\mathbf{R}_i}$ is controllable with respect to $\|_{i=1}^n P_{i+\Sigma_i}(L)$, hence with respect to $L \subseteq \|_{i=1}^n P_{i+\Sigma_i}(L)$, and A_{uc}. Using transitivity of controllability, Lemma 19 in the appendix, we obtain that $\|_{i=1}^n (\mathbf{R}_i \parallel L_C)$ is controllable with respect to L and A_{uc}.

Since $\mathbf{R}_i \parallel L_C$ is trivially observable, Theorem 1 implies that $\|_{i=1}^n (\mathbf{R}_i \parallel L_C)$ is coobservable with respect to L and $(A_{o,i} \cup \Sigma_i \cup \Sigma')_{i=1}^n$. It again holds under the assumption that $(A_{o,i} \cup \Sigma_i \cup \Sigma') \cap A_c \subseteq (A_{c,i} \cup \Sigma_{c,i} \cup \Sigma'_c)$.

It remains to show the other inclusion $\sup \mathrm{C} \subseteq \|_{i=1}^n (\mathbf{R}_i \parallel L_C)$. We show it in two steps. First we show that $\sup \mathrm{C} \subseteq \|_{i=1}^n \mathbf{R}_i$ and then that $P_{\Sigma'}(\sup \mathrm{C}) \subseteq L_C$.

To show that $\sup \mathrm{C} \subseteq \|_{i=1}^n \mathbf{R}_i$, we prove that, for every i, $P_{i+\Sigma_i}(\sup \mathrm{C}) \subseteq \mathbf{R}_i$ by showing that $P_{i+\Sigma_i}(\sup \mathrm{C}) \subseteq P_{i+\Sigma_i}(K)$ is controllable with respect to $P_{i+\Sigma_i}(L)$. To this end, let $t \in \overline{P_{i+\Sigma_i}(\sup \mathrm{C})}$, $a \in (A_{o,i} \cup \Sigma_i) \cap A_{uc}$, and $ta \in P_{i+\Sigma_i}(L)$. Then, there exists $s \in \overline{\sup \mathrm{C}}$ such that $P_{i+\Sigma_i}(s) = t$. Since $P_{i+\Sigma_i}$ is an L-observer, there exists $v \in A^*$ such that $sv \in L$ and $P_{i+\Sigma_i}(sv) = ta$, that is, $v = ua$ for some $u \in (A \setminus (A_{o,i} \cup \Sigma_i))^*$. The LCC property of $P_{i+\Sigma_i}$ for L and $sua \in L$ imply that there exists

$u' \in (A_{uc} \setminus (A_{o,i} \cup \Sigma_i))^*$ such that $su'a \in L$. Since u' is uncontrollable, controllability of $\sup \mathrm{C}$ with respect to L and A_{uc} implies that $su'a \in \overline{\sup \mathrm{C}}$, that is, $P_{i+\Sigma_i}(su'a) = ta \in \overline{P_{i+\Sigma_i}(\sup \mathrm{C})}$. Thus, $P_{i+\Sigma_i}(\sup \mathrm{C})$ is controllable with respect to $P_{i+\Sigma_i}(L)$, hence $P_{i+\Sigma_i}(\sup \mathrm{C}) \subseteq \mathbf{R}_i$.

Finally, we show that $P_{\Sigma'}(\sup \mathrm{C}) \subseteq L_C$, i.e., that it is a subset of $\|_{i=1}^n P_{\Sigma'}(\mathbf{R}_i)$ controllable with respect to $\|_{i=1}^n \overline{P_{\Sigma'}(\mathbf{R}_i)}$ and Σ'_{uc}. Since $P_{i+\Sigma_i}(\sup \mathrm{C}) \subseteq \mathbf{R}_i$, we have that $\sup \mathrm{C} \subseteq \|_{i=1}^n P_{i+\Sigma_i}(\sup \mathrm{C}) \subseteq \|_{i=1}^n \mathbf{R}_i$. Applying projection $P_{\Sigma'}$, we obtain $P_{\Sigma'}(\sup \mathrm{C}) \subseteq P_{\Sigma'}(\|_{i=1}^n \mathbf{R}_i) = \|_{i=1}^n P_{\Sigma'}(\mathbf{R}_i)$, where the last equality is by Lemma 18, see the appendix. Using Theorem 2 in Pena et al. [29] saying that a composition of observers is an observer, the assumption on $P_{\Sigma'}$ to be an $\mathbf{R}_i$-observer, which also means that $P_{\Sigma'}$ is an $\overline{\mathbf{R}_i}$-observer, implies that $P_{\Sigma'}$ is a $(\|_{i=1}^n \overline{\mathbf{R}_i})$-observer. We show that $P_{\Sigma'}(\sup \mathrm{C})$ is controllable with respect to $\|_{i=1}^n \overline{P_{\Sigma'}(\mathbf{R}_i)}$ and Σ'_{uc}. To this end, let $t \in \overline{P_{\Sigma'}(\sup \mathrm{C})}$, $a \in \Sigma'_{uc}$, and $ta \in \|_{i=1}^n \overline{P_{\Sigma'}(\mathbf{R}_i)} = P_{\Sigma'}(\|_{i=1}^n \overline{\mathbf{R}_i})$. Then, there exists $s \in \overline{\sup \mathrm{C}}$ such that $P_{\Sigma'}(s) = t$. Since $P_{\Sigma'}$ is a $(\|_{i=1}^n \overline{\mathbf{R}_i})$-observer, there exists $v \in A^*$ such that $sv \in \|_{i=1}^n \overline{\mathbf{R}_i}$ and $P_{\Sigma'}(sv) = ta$, that is, $v = ua$ for some $u \in (A \setminus \Sigma')^*$. The LCC property of $P_{\Sigma'}$ for $\|_{i=1}^n \overline{\mathbf{R}_i}$ and $sua \in \|_{i=1}^n \overline{\mathbf{R}_i}$ then imply that there exists $u' \in (A_{uc} \setminus \Sigma')^*$ such that $su'a \in \|_{i=1}^n \overline{\mathbf{R}_i}$. Since $\|_{i=1}^n \overline{\mathbf{R}_i} \subseteq L$, and u' is uncontrollable, controllability of $\sup \mathrm{C}$ with respect to L and A_{uc} implies that $su'a \in \overline{\sup \mathrm{C}}$. That is, $P_{\Sigma'}(su'a) = ta \in \overline{P_{\Sigma'}(\sup \mathrm{C})}$, hence $P_{\Sigma'}(\sup \mathrm{C}) \subseteq L_C$.

Together, $\sup \mathrm{C} = \|_{i=1}^n \mathbf{R}_i \parallel L_C = \|_{i=1}^n (\mathbf{R}_i \parallel L_C)$, which completes the proof. □

Notice that it is sufficient to require that $P_{\Sigma'}$ is an L-observer and LCC for L.

In case of partial observations, one could obtain the global supremal controllable and normal sublanguage in a similar way under the assumptions similar to those discussed below Theorem 11. Since, as already mentioned, there is no global optimal solution in the case of partial observations, we do not discuss this case in more detail.

5.5 Complexity

We now briefly discuss the complexity of our approach. Since we use standard notions, the complexity mainly depends on the complexity of corresponding algorithms for the computation of controllable and observable languages.

In Assumption 6, we assume that K is separable. If this is not the case, a polynomial-time algorithm [15] is used to find extensions $(\Sigma_i)_{i=1}^n$ making the language separable with respect to extended alphabets. Then we compute local supervisors $\mathbf{R}_i$ using the standard algorithms discussed in Subsection 5.2. This requires to compute the projections of K and L, which is exponential in the worst case. However, the alphabet Σ_i can be chosen so that the projection $P_{i+\Sigma_i}$ is K- and L-observer, which then results in the computation of those projections in polynomial time [51]. The computation of $\mathbf{R}_i$ is then performed in the respective time. It is polynomial in the case of full observations. To check/ensure optimality in the case of full observations, we either check whether the polynomially many pairs of languages are mutually controllable, which can be done in polynomial time, or further extend the alphabets Σ_i in polynomial time so that the observer and LCC conditions of Theorem 11 are satisfied. Furthermore, Theorems 7 and 11 require that the computed supervisors are nonconflicting. This is a PSPACE-complete problem [35]. However, this test can be skipped and we can directly compute the language L_C from Theorem 12, which may, in the worst case, require exponential space with respect to the number of local supervisors.

6 Conclusion

In this paper, we have shown how to construct a solution to the decentralized control problem (a controllable and coobservable sublanguage of a specification) by using additional communications. Our approach relies on the notion of conditional decomposability recently studied by the authors, which overcomes the undecidable problem to find a separable sublanguage of the specification. The computation of local supervisors is fully decentralized and coobservability is guaranteed by construction. We discussed two ways how to obtain the globally optimal solution in case of full observations if the computed languages are synchronously nonconflicting (prefix-closed). One is based on the notion of mutual controllability, the other on increasing the communication between supervisors. Indeed, both approaches can be combined as preferred. Our approach can be used for both prefix-closed and non-prefix-closed specifications. For conflicting supervisors, we showed how to impose nonconflictness, hence coobservability.

A Proof of Theorem 2

Theorem 2. *The following problem is PSPACE-complete.*

INPUT: *Alphabets $E_1, E_2, \ldots, E_n$ and a generator H over $\cup_{i=1}^n E_i$.*

OUTPUT: `Yes` *if and only if $L_m(H)$ is separable with respect to $(E_i)_{i=1}^n$.*

Proof. Standard techniques simulating a product automaton on-the-fly show that it belongs to PSPACE. To prove hardness, we reduce the finite-state automata intersection problem (INT): Given a set of deterministic finite automata $\{G_i\}_{i=1}^n$ over a common alphabet Σ. Is $\bigcap_{i=1}^n L_m(G_i) = \emptyset$? The problem is PSPACE-complete [22].

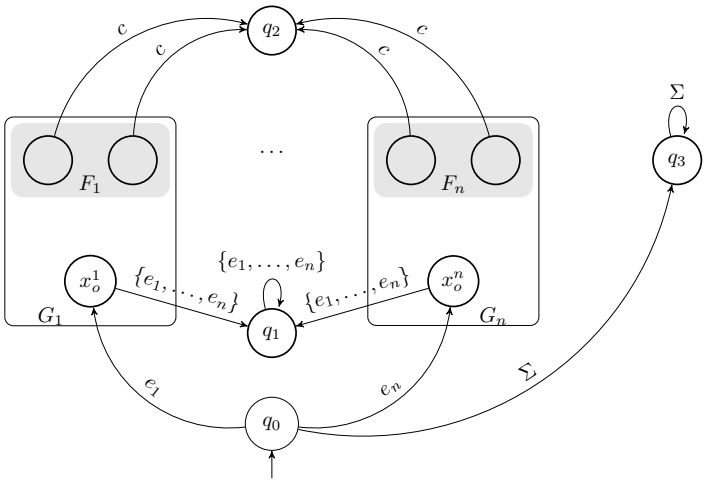

Figure 2: The automaton H

Let a set of deterministic automata $\{G_i\}_{i=1}^n$ with a common alphabet Σ be an instance of INT. Without loss of generality, we assume that $n \geq 3$, since if n is constant, then the problem is solvable in PTIME. Let $G_i = (X_i, \Sigma, \delta_i, x_o^i, F_i)$ and assume that all states of G_i are reachable from the initial state x_o^i. This assumption does not change the complexity. We construct a deterministic automaton H and alphabets $(E_i)_{i=1}^n$ in polynomial time such that $L(H)$ is separable if and only if $\bigcap_{i=1}^n L_m(G_i) = \emptyset$.

To this end, we define the automaton $H = (X, E, \delta, q_0, X)$ so that the set of states is $X = \bigcup_{i=1}^n X_i \cup \{q_0, q_1, q_2, q_3\}$, where q_0, q_1, q_2, q_3 are new states, $E = \Gamma \cup \Sigma$, where $\Gamma = \{e_1, \ldots, e_n, c\}$ is an alphabet such that $\Gamma \cap \Sigma = \emptyset$, and the transition function is defined as follows. The initial state q_0 goes under e_i to the initial state x_o^i of G_i, $i = 1, \ldots, n$, and for every a in Σ, q_0 goes under a to q_3. In q_3, there is a self-loop under every a in Σ. The transitions inside every G_i are unchanged. For every e in $\{e_1, \ldots, e_n\}$ and every $i = 1, \ldots, n$, we add a transition from x_o^i to q_1 under e. State q_1 contains a self-loop for every e in $\{e_1, \ldots, e_n\}$. Finally, for $i = 1, \ldots, n$, we add a c-transition from all states of F_i of G_i to state q_2, cf. Fig. 2.

To complete the reduction, we define $E_i = E \setminus \{e_i\}$, which defines the projection $P_i \colon E^* \to E_i^*$, $i = 1, \ldots, n$. Note that the reduction is polynomial. We show that $L(H)$ is separable with respect to $(E_i)_{i=1}^n$ if and only if $\bigcap_{i=1}^n L_m(G_i) = \emptyset$.

Assume that $t \in \bigcap_{i=1}^n L_m(G_i)$. Then $e_i t c \in L(H)$, for all $i = 1, \ldots, n$, which implies that $tc \in P_i^{-1}(P_i(e_i t c))$, hence $tc \in \|_{i=1}^n P_i(L(H))$. However, $tc \notin L(H)$, which shows that $L(H)$ is not separable with respect to $(E_i)_{i=1}^n$.

To prove the other direction, we assume that $L(H)$ is not separable and show that $\bigcap_{i=1}^n L_m(G_i) \neq \emptyset$. Let $w \in \|_{i=1}^n P_i(L(H))$ and $w \notin L(H)$. Note that $\Sigma^* \cup \{e_1, \ldots, e_n\}^* \subseteq L(H)$ because $L(H) = \Sigma^* \cup \{e_1, \ldots, e_n\}^* \cup \bigcup_{i=1}^n e_i L_m(G_i) c \cup \bigcup_{i=1}^n e_i L(G_i)$. Therefore, we have that for $i = 1, \ldots, n$, the word w belongs to $P_i^{-1} P_i(L(H)) = (\Sigma \cup \{e_i\})^* \cup \{e_1, \ldots, e_n\}^* \cup P_i^{-1}(L_m(G_i)c) \cup \bigcup_{j \neq i} P_i^{-1}(e_j L_m(G_j) c) \cup P_i^{-1}(L(G_i)) \cup \bigcup_{j \neq i} P_i^{-1}(e_j L(G_j))$.

We first show that if c does not appear in w, then w belongs to $L(H)$. In this case, based on the above observation, w must contain at least one event from Σ and at least one event from $\{e_1, \ldots, e_n\}$. Thus,

$$w \in (\Sigma \cup \{e_i\})^* \cup \bigcup_{j \neq i} P_i^{-1}(e_j L(G_j))$$

for $i = 1, \ldots, n$, because $P_i^{-1}(L(G_i)) \subseteq (\Sigma \cup \{e_i\})^*$. Then $w \in \bigcap_{i=1}^n T_i$, where T_i is one of the languages forming the union above.

If, for some $i \neq j$, $T_i = (\Sigma \cup \{e_i\})^*$ and $T_j = (\Sigma \cup \{e_j\})^*$, then $w \in \Sigma^* \subseteq L(H)$; a contradiction.

If there is only one i such that $T_i = (\Sigma \cup \{e_i\})^*$ and, for all $j \neq i$, $T_j = P_j^{-1}(e_{k_j} L(G_{k_j}))$, where $k_j \neq j$, then w belongs to $\bigcap_{\ell=1}^n T_\ell$ if and only if $e_{k_j} = e_i$, for all $j \neq i$. Hence, for $j \neq i$, $T_i \cap T_j = e_i L(G_i)$, which implies that $\bigcap_{\ell=1}^n T_\ell = e_i L(G_i) \subseteq L(H)$; a contradiction.

The last option is that, for all i, $T_i = P_i^{-1}(e_{j_i} L(G_{j_i}))$, where $j_i \neq i$. Then there exist $j_k \neq j_\ell$ such that $e_{j_k} \neq e_{j_\ell}$. Without loss of generality, we assume that $w = v_1 e_{j_k} v_2 e_{j_\ell} w'$, where $v_1 v_2 w' \in (\Sigma \cup \{e_1, \ldots, e_n\})^*$. Since $w \in T_k = P_k^{-1}(e_{j_k} L(G_{j_k}))$, $P_k(w) \in e_{j_k} L(G_{j_k})$, hence $P_k(v_1) = \varepsilon$ and $P_k(v_2 e_{j_\ell} w') \in \Sigma^*$, which implies that $j_\ell = k$, $v_1 \in \{e_k\}^*$ and $v_2 w' \in (\Sigma \cup \{e_k\})^*$. Similarly, $P_\ell(w) \in e_{j_\ell} L(G_{j_\ell})$ implies that $j_k = \ell$, $v_1 v_2 \in \{e_\ell\}^*$ and $w' \in (\Sigma \cup \{e_\ell\})^*$. Together, $v_1 v_2 = \varepsilon$, $w' \in \Sigma^*$, and $w = e_\ell e_k w'$, for $k \neq \ell$. By the assumption, there is a projection P_m such that $P_m \notin \{P_k, P_\ell\}$. Since $w \in T_m$, $P_m(w) \in (\Sigma \cup \{e_{j_m}\})^*$, and $P_m(e_\ell) = \varepsilon$ or $P_m(e_k) = \varepsilon$. The first case gives that $P_m = P_\ell$, the second that $P_m = P_k$, which is a contradiction.

Thus, we have show that if c does not appear in w, then w belongs to $L(H)$.

Assume that c appears in w. By the analysis above, $w \in \bigcap_{i=1}^n [P_i^{-1}(L_m(G_i)c) \cup \bigcup_{j \neq i} P_i^{-1}(e_j L_m(G_j) c)]$. It implies that there is exactly one c in w. Again, $w \in \bigcap_{i=1}^n T_i$, where T_i is one of the elements of the union.

Analogously as above, if, for all i, $T_i = P_i^{-1}(e_{j_i} L_m(G_{j_i}) c)$, where $j_i \neq i$, then there are $j_k \neq j_\ell$ such that $e_{j_k} \neq e_{j_\ell}$. Without loss of generality, let $w = v_1 e_{j_k} v_2 e_{j_\ell} w' c w''$, where $v_1 v_2 w' w'' \in (\Sigma \cup \{e_1, \ldots, e_n\})^*$. Then $P_k(w) \in e_{j_k} L_m(G_{j_k}) c$,

hence we have that $P_k(v_1) = \varepsilon$ and $P_k(v_2 e_{j_\ell} w' w'') \in \Sigma^*$, which implies that $j_\ell = k$, $v_1 \in \{e_k\}^*$ and $v_2 w' w'' \in (\Sigma \cup \{e_k\})^*$. Similarly, $P_\ell(w) \in e_{j_\ell} L_m(G_{j_\ell})c$ implies that $j_k = \ell$, $v_1 v_2 \in \{e_\ell\}^*$ and $w' w'' \in (\Sigma \cup \{e_\ell\})^*$. Together, $v_1 v_2 = \varepsilon$, $w' w'' \in \Sigma^*$, and $w = e_\ell e_k w' c w''$, for $k \neq \ell$. Let $P_m \notin \{P_k, P_\ell\}$ be a projection. Since $P_m(w) \in (\Sigma \cup \{e_{j_m}, c\})^*$, $P_m(e_\ell) = \varepsilon$ or $P_m(e_k) = \varepsilon$. The first case gives that $P_m = P_\ell$, the second that $P_m = P_k$, which is a contradiction.

Thus, there must exist i such that $T_i = P_i^{-1}(L_m(G_i)c)$. Then $P_i(w) \in L_m(G_i)c$, which implies that $w \in (\Sigma \cup \{e_i, c\})^*$. This means that, for $j \neq i$, $P_j(w) = w \in L_m(G_j)c \cup e_i L_m(G_i)c$. If $T_j = e_i L_m(G_i)c$, for some $j \neq i$, then $w \in (\Sigma \cup \{e_i, c\})^* \cap e_i L_m(G_i)c = e_i L_m(G_i)c \subseteq L(H)$; a contradiction again. Thus, it must be that for every $j \neq i$, $T_j = L_m(G_j)c$. Then $w \in \bigcap_{i=1}^n T_i = \bigcap_{i=1}^n L_m(G_i)c$ implies that $\bigcap_{i=1}^n L_m(G_i) \neq \emptyset$. □

B Auxiliary Results

Lemma 14 ([8]). *For $i = 1, 2$, let L_i be a prefix-closed language over A_i, and let $K_i \subseteq L_i$ be controllable with respect to L_i and $A_{i,uc}$. Let $A = A_1 \cup A_2$. If K_1 and K_2 are synchronously nonconflicting, then $K_1 \parallel K_2$ is controllable with respect to $L_1 \parallel L_2$ and A_{uc}.*

Lemma 15. *For $i = 1, 2$, let L_i be a prefix-closed language over A_i, and let $K_i \subseteq L_i$ be normal with respect to L_i, $A_{i,uc}$ and $P_{o,i} \colon A_i^* \to A_{o,i}^*$. Let $A = A_1 \cup A_2$. If K_1 and K_2 are synchronously nonconflicting, then $K_1 \parallel K_2$ is normal with respect to $L_1 \parallel L_2$, A_{uc} and $P_o \colon A^* \to A_o^*$.*

Proof. $P_o^{-1} P_o(\overline{K_1 \parallel K_2}) \cap L_1 \parallel L_2 \subseteq P_{o,1}^{-1} P_{o,1}(\overline{K_1}) \parallel P_{o,2}^{-1} P_{o,2}(\overline{K_2}) \parallel L_1 \parallel L_2 = \overline{K_1} \parallel \overline{K_2} = \overline{K_1 \parallel K_2}$. As the other inclusion always holds, the proof is complete. □

Lemma 16. *For $i = 1, 2$, let L_i be a prefix-closed language over A_i, and let $K_i \subseteq L_i$ be observable with respect to L_i, $A_{i,uc}$ and $P_{o,i} \colon A_i^* \to A_{o,i}^*$. Let $A = A_1 \cup A_2$. If K_1 and K_2 are synchronously nonconflicting, then $K_1 \parallel K_2$ is observable with respect to $L_1 \parallel L_2$, A_{uc} and $P_o \colon A^* \to A_o^*$.*

Proof. Let $s, s' \in A^*$ be such that $P_o(s) = P_o(s')$. Let $a \in A$ and assume that $sa \in \overline{K_1 \parallel K_2}$, $s' \in \overline{K_1 \parallel K_2}$, and $s'a \in L_1 \parallel L_2$. Let $P_i \colon A^* \to A_i^*$, for $i = 1, 2$. Then $P_i(sa) \in \overline{K_i}$, $P_i(s') \in \overline{K_i}$, and $P_i(s'a) \in L_i$ imply that $P_i(s'a) \in \overline{K_i}$, by observability of K_i with respect to L_i. Thus, $s'a \in \overline{K_1} \parallel \overline{K_2} = \overline{K_1 \parallel K_2}$. □

Lemma 17 ([29]). *Let $L_i \subseteq A_i^*$, $i \in J$, and $\bigcup_{k,\ell \in J}^{k \neq \ell} (A_k \cap A_\ell) \subseteq A_0$. If $P_{i,0} \colon A_i^* \to (A_i \cap A_0)^*$ is an L_i-observer, for $i \in J$, then $\overline{\|_{i \in J} L_i} = \|_{i \in J} \overline{L_i}$ if and only if $\overline{\|_{i \in J} P_{i,0}(L_i)} = \|_{i \in J} \overline{P_{i,0}(L_i)}$.*

Lemma 18 ([52]). *Let $P_k \colon A^* \to A_k^*$ be a projection, and let $L_i \subseteq A_i^*$, where $A_i \subseteq A$, for $i = 1, 2$, and $A_1 \cap A_2 \subseteq A_k$. Then $P_k(L_1 \| L_2) = P_k(L_1) \| P_k(L_2)$.*

Lemma 19 ([16]). *Let $K \subseteq L \subseteq M$ be such that K is controllable with respect to $\overline{L}$ and L is controllable with respect to $\overline{M}$. Then K is controllable with respect to $\overline{M}$.*

Acknowledgements

The authors are grateful to the anonymous reviewers, whose comments and suggestions significantly improved the paper. The work was supported by RVO 67985840, by GAČR in project GA15-02532S and by the German Research Foundation (DFG) in Emmy Noether grant KR 4381/1-1 (DIAMOND).

References

[1] G. Barrett and S. Lafortune. Decentralized supervisory control with communicating controllers. *IEEE Transactions on Automatic Control*, 45(9):1620–1638, 2000.

[2] R. D. Brandt, V. Garg, R. Kumar, F. Lin, S. I. Marcus, and W. M. Wonham. Formulas for calculating supremal controllable and normal sublanguages. *Systems & Control Letters*, 15(2):111–117, 1990.

[3] H.J. Bravo, A.E.C. Da Cunha, P.N. Pena, R. Malik, and J.E.R. Cury. Generalised verification of the observer property in discrete event systems. In *WODES*, pages 337–342, Mexico, 2012.

[4] Kai Cai, Renyuan Zhang, and W. Murray Wonham. Relative observability of discrete-event systems and its supremal sublanguages. *IEEE Transactions on Automatic Control*, 60(3):659–670, 2015.

[5] C.G. Cassandras and S. Lafortune. *Introduction to discrete event systems*. Springer, second edition, 2008.

[6] H. Chakib and A. Khoumsi. Multi-decision supervisory control: Parallel decentralized architectures cooperating for controlling discrete event systems. *IEEE Transactions on Automatic Control*, 56(11):2608–2622, 2011.

[7] Jinghuai Fa, Xiaojun Yang, and Yingping Zheng. Formulas for a class of controllable and observable sublanguages larger than the supremal controllable and normal sublanguage. *Systems & Control Letters*, 20(1):11–18, 1993.

[8] L. Feng. *Computationally Efficient Supervisor Design for Discrete-Event Systems*. PhD thesis, University of Toronto, Ontario, Canada, 2007.

[9] L. Feng and W.M. Wonham. On the computation of natural observers in discrete-event systems. *Discrete Event Dynamic Systems*, 20(1):63–102, 2010.

[10] S. Jiang and R. Kumar. Decentralized control of discrete event systems with specializations to local control and concurrent systems. *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics*, 30(5):653–660, 2000.

[11] S. Jiang, R. Kumar, and H. E. Garcia. Optimal sensor selection for discrete-event systems with partial observation. *IEEE Transactions on Automatic Control*, 48(3):369–381, 2003.

[12] J. Komenda, H. Marchand, and S. Pinchinat. A constructive and modular approach to decentralized supervisory control problems. *IFAC Proceedings Volumes*, 39(17):111–116, 2006. 3rd IFAC Workshop on Discrete-Event System Design.

[13] J. Komenda and T. Masopust. A bridge between decentralized and coordination control. In *Allerton Conference on Communication, Control, and Computing*, pages 966–972, 2013.

[14] J. Komenda, T. Masopust, and J.H. van Schuppen. Synthesis of controllable and normal sublanguages for discrete-event systems using a coordinator. *Systems & Control Letters*, 60(7):492–502, 2011.

[15] J. Komenda, T. Masopust, and J.H. van Schuppen. On conditional decomposability. *Systems & Control Letters*, 61(12):1260–1268, 2012.

[16] J. Komenda, T. Masopust, and J.H. van Schuppen. Supervisory control synthesis of discrete-event systems using a coordination scheme. *Automatica*, 48(2):247–254, 2012.

[17] J. Komenda, T. Masopust, and J.H. van Schuppen. Coordination control of discrete-event systems revisited. *Discrete Event Dynamic Systems*, 25(1-2):65–94, 2015.

[18] J. Komenda, T. Masopust, and J.H. van Schuppen. Relative observability in coordination control. In *IEEE International Conference on Automation Science and Engineering (CASE)*, pages 75–80, 2015.

[19] Jan Komenda and Jan H. van Schuppen. Modular antipermissive control of discrete-event systems. *IFAC World Congress*, 38(1):97–102, 2005.

[20] Jan Komenda and Jan H. van Schuppen. Modular control of discrete-event systems with coalgebra. *IEEE Transactions on Automatic Control*, 53(2):447–460, 2008.

[21] P. Kozák and W. M. Wonham. Fully decentralized solutions of supervisory control problems. *IEEE Transactions on Automatic Control*, 40(12):2094–2097, 1995.

[22] D. Kozen. Lower bounds for natural proof systems. In *FOCS*, pages 254–266, 1977.

[23] R. Kumar and S. Takai. Inference-based ambiguity management in decentralized decision-making: Decentralized control of discrete event systems. *IEEE Transactions on Automatic Control*, 52(10):1783–1794, 2007.

[24] Ratnesh Kumar and Vijay K. Garg. Optimal supervisory control of discrete event dynamical systems. *SIAM Journal on Control and Optimization*, 33(2):419–439, 1995.

[25] S. Lafortune and E. Chen. The infimal closed controllable superlanguage and its application in supervisory control. *IEEE Transactions on Automatic Control*, 35(4):398–405, 1990.

[26] S.-H. Lee and K.C. Wong. Structural decentralized control of concurrent discrete-event systems. *European Journal of Control*, 8(5):477–491, 2002.

[27] L. Lin, A. Stefanescu, R. Su, W. Wang, and A.R. Shehabinia. Towards decentralized synthesis: Decomposable sublanguage and joint observability problems. In *American Control Conference*, pages 2047–2052, 2014.

[28] R. Malik. Programming a fast explicit conflict checker. In *WODES*, pages 438–443, 2016.

[29] P.N. Pena, J.E.R. Cury, and S. Lafortune. Verification of nonconflict of supervisors using abstractions. *IEEE Transactions on Automatic Control*, 54(12):2803–2815, 2009.

[30] P.N. Pena, J.E.R. Cury, R. Malik, and S. Lafortune. Efficient computation of observer projections using OP-verifiers. In *WODES*, pages 416–421, 2010.

[31] P.J. Ramadge and W.M. Wonham. Supervisory control of a class of discrete event processes. *SIAM Journal on Control and Optimization*, 25(1):206–230, 1987.

[32] S. L. Ricker and K. Rudie. Knowledge is a terrible thing to waste: Using inference in discrete-event control problems. *IEEE Transactions on Automatic Control*, 52(3):428–441, 2007.

[33] S.L. Ricker and K. Rudie. Know means no: Incorporating knowledge into discrete-event control systems. *IEEE Transactions on Automatic Control*, 45(9):1656–1668, 2000.

[34] K. Rohloff, S. Khuller, and G. Kortsarz. Approximating the minimal sensor selection for supervisory control. *Discrete Event Dynamic Systems*, 16(1):143–170, 2006.

[35] K. Rohloff and S. Lafortune. PSPACE-completeness of modular supervisory control problems. *Discrete Event Dynamic Systems*, 15:145–167, 2005.

[36] K. Rohloff, T.-S. Yoo, and S. Lafortune. Deciding co-observability is PSPACE-complete. *IEEE Transactions on Automatic Control*, 48(11):1995–1999, 2003.

[37] K. Rudie, S. Lafortune, and F. Lin. Minimal communication in a distributed discrete-event system. *IEEE Transactions on Automatic Control*, 48(6):957–975, 2003.

[38] K. Rudie and W. M. Wonham. Supervisory control of communicating processes. In *Protocol Specification, Testing and Verification X*, pages 243–257, 1990.

[39] K. Rudie and W.M. Wonham. Think globally, act locally: Decentralized supervisory control. *IEEE Transactions on Automatic Control*, 37(11):1692–1708, 1992.

[40] Karen Rudie and W. Murray Wonham. The infimal prefix-closed and observable superlanguage of given language. *Systems & Control Letters*, 15(5):361–371, 1990.

[41] K. Schmidt and C. Breindl. On maximal permissiveness of hierarchical and modular supervisory control approaches for discrete event systems. In *WODES*, pages 462–467, 2008.

[42] K. Schmidt and C. Breindl. Maximally permissive hierarchical control of decentralized discrete event systems. *IEEE Transactions on Automatic Control*, 56(4):723–737, 2011.

[43] S. Takai. On the language generated under fully decentralized supervision. *IEEE Transactions on Automatic Control*, 43(9):1253–1256, 1998.

[44] S. Takai and R. Kumar. Synthesis of inference-based decentralized control for discrete event systems. *IEEE Transactions on Automatic Control*, 53(2):522–534, 2008.

[45] Shigemasa Takai and Toshimitsu Ushio. Effective computation of an $L_m(G)$-closed, controllable, and observable sublanguage arising in supervisory control. *Systems & Control Letters*, 49(3):191–200, 2003.

[46] J.G. Thistle. Undecidability in decentralized supervision. *Systems & Control Letters*, 54(5):503–509, 2005.

[47] S. Tripakis. Undecidable problems of decentralized observation and control on regular languages. *Information Processing Letters*, 90(1):21–28, 2004.

[48] W. Wang, A. R. Girard, S. Lafortune, and F. Lin. On codiagnosability and coobservability with dynamic observations. *IEEE Transactions on Automatic Control*, 56(7):1551–1566, 2011.

[49] W. Wang, S. Lafortune, and F. Lin. Optimal sensor activation in controlled discrete event systems. In *Conference on Decision and Control*, pages 877–882. IEEE, 2008.

[50] Y. Willner and M. Heymann. Supervisory control of concurrent discrete-event systems. *International Journal of Control*, 54(5):1143–1169, 1991.

[51] K. Wong. On the complexity of projections of discrete-event systems. In *WODES*, pages 201–206, 1998.

[52] W.M. Wonham. Supervisory control of discrete-event systems. University of Toronto, 2012. Available at http://www.control.utoronto.ca/DES/.

[53] X. Yin and S. Lafortune. Synthesis of maximally permissive supervisors for partially-observed discrete-event systems. *IEEE Transactions on Automatic Control*, 61(5):1239–1254, 2016.

[54] T.S. Yoo and S. Lafortune. A general architecture for decentralized supervisory control of discrete-event systems. *Discrete Event Dynamic Systems*, 12(3):335–377, 2002.

[55] T.S. Yoo and S. Lafortune. Decentralized supervisory control with conditional decisions: Supervisor existence. *IEEE Transactions on Automatic Control*, 49(11):1886–1904, 2004.

[56] H. Zhong and W. M. Wonham. On the consistency of hierarchical supervision in discrete-event systems. *IEEE Transactions on Automatic Control*, 35(10):1125–1134, 1990.