Coordination control of discrete-event systems revisited

Jan Komenda · Tomáš Masopust · Jan H. van Schuppen

Received: 30 January 2013 / Accepted: 6 December 2013 / Published online: 13 February 2014
© Springer Science+Business Media New York 2014

Abstract In this paper, we revise and further investigate the coordination control approach proposed for supervisory control of distributed discrete-event systems with synchronous communication based on the Ramadge-Wonham automata framework. The notions of conditional decomposability, conditional controllability, and conditional closedness ensuring the existence of a solution are carefully revised and simplified. The approach is generalized to non-prefix-closed languages, that is, supremal conditionally controllable sublanguages of not necessary prefix-closed languages are discussed. Non-prefix-closed languages introduce the blocking issue into coordination control, hence a procedure to compute a coordinator for nonblockingness is included. The optimization problem concerning the size of a coordinator is under investigation. We prove that to find the minimal extension of the coordinator event set for which a given specification language is conditionally decomposable is NP-hard. In other words, unless P=NP, it is not possible to find a polynomial algorithm to compute the minimal coordinator with respect to the number of events.

Keywords Discrete-event systems · Distributed systems with synchronous communication · Supervisory control · Coordination control · Conditional decomposability

A preliminary version was presented at the 11th International Workshop on Discrete Event Systems (WODES 2012) held in Guadalajara, Mexico (Komenda et al. 2012a).

J. Komenda · T. Masopust (✉)
Institute of Mathematics, Academy of Sciences of the Czech Republic, Žižkova 22, 616 62 Brno, Czech Republic
e-mail: masopust@ipm.cz

J. Komenda
e-mail: komenda@ipm.cz

J. H. van Schuppen
Van Schuppen Control Research, Gouden Leeuw 143, 1103 KB Amsterdam, The Netherlands
e-mail: jan.h.van.schuppen@xs4all.nl
1 Introduction

In this paper, we revise and further investigate the coordination control approach proposed for supervisory control of distributed discrete-event systems with synchronous communication based on the Ramadge-Wonham automata framework. A distributed discrete-event system with synchronous communication is modeled as a parallel composition of two or more subsystems, each of which has its own observation channel. The local control synthesis consists in synthesizing local nonblocking supervisors for each of the subsystems. It is well-known that such a purely decentralized (often referred to as modular) approach does not work in general. Recently, Komenda and van Schuppen (2008) have proposed a coordination control architecture as a trade-off between the purely local control synthesis, which is not effective in general because the composition of local supervisors may violate the specification, and the global control synthesis, which is not always possible because of the complexity reasons since the composition of all subsystems can result in an exponential blow-up of states in the monolithic plant. The coordination control approach has been developed for prefix-closed languages in Komenda et al. (2012c) and extended to systems with partial observations in Komenda et al. (2011b). The case of non-prefix-closed languages has partially been discussed in Komenda et al. (2011a). Most of these approaches for prefix-closed languages have already been implemented in the software library libFAUDES (Moor et al. 2012).

In the last two decades several alternative approaches have been proposed for supervisory control of large discrete-event systems. Among the different control architectures are such as hierarchical control based on abstraction (Schmidt et al. 2008; Wong and Wonham 1996; Zhong and Wonham 1990), modular approaches (Feng 2007; Gaudin and Marchand 2004; Komenda et al. 2008; Pena et al. 2009), decentralized control (Rudie and Wonham 1992; Yoo and Lafortune 2002) also with inferencing (conditional decisions) (Kumar and Takai 2007; Yoo and Lafortune 2004) or with communicating supervisors (Ricker and Rudie 2000), and the so-called interface-based approach (Leduc et al. 2005). Nowadays, these approaches are combined to achieve even better results, cf. (Schmidt and Breindl 2008; 2011). Our coordination control approach can be seen as a combination of the horizontal and vertical modularity. The coordinator level corresponds to the abstraction (i.e., the higher level) of hierarchical control, while the local control synthesis is a generalization of the modular control synthesis. Moreover, coordination control is closely related to decentralized control with communication, because local supervisors communicate indirectly via a coordinator, cf. (Barrett and Lafortune 2000).

In this paper, the notions of conditional decomposability, conditional controllability, and conditional closedness, which are the central notions to characterize the solvability of the coordination control problem, are carefully revised and simplified. The coordination control approach is generalized to non-prefix-closed languages, hence supremal conditionally controllable sublanguages of not necessary prefix-closed languages are discussed. This generality, however, introduces the problem of nonblockingness into the coordination control approach, therefore a part with a procedure to compute a coordinator for nonblockingness is included in the paper. The optimization problem concerning the size of a coordinator is nowadays the main problem under investigation. The construction of a coordinator described in this paper depends mainly on a set of events, including the set of all shared events. We prove that to construct the coordinator so that its event set is minimal with respect to the number of events or, in other words, to find the minimal extension of the coordinator event set for which a given specification language is conditionally decomposable, is NP-hard.
The main contributions and the organization of the paper are as follows. Section 2 recalls the basics of supervisory control theory and revises the fundamental concepts. Section 3 gives the computational complexity analysis of the minimal extension problem for conditional decomposability and proves that it is NP-hard to find the minimal extension with respect to set inclusion (Corollary 1). Section 4 formulates the problem of coordination supervisory control. The notion of conditional controllability (Definition 3) is revised and simplified, however still equivalent to the previous definition in, e.g., Komenda et al. (2012c). Section 5 provides results concerning non-prefix-closed languages. Theorem 6 shows that in a special case the parallel composition of local supervisors results in the supremal conditionally controllable languages. However, the problem how to compute the supremal conditionally controllable sublanguage in general is open. Section 6 discusses the construction of a coordinator for nonblockingness (Theorem 8) and presents an algorithm. Section 7 revises the prefix-closed case, where a less restrictive condition, LCC, is used instead of OCC. The possibility to use LCC instead of OCC has already been mentioned in Komenda et al. (2012c) without proofs, therefore the proofs are provided here. Finally, Section 8 concludes the paper.

2 Preliminaries and definitions

We assume that the reader is familiar with the basic notions and concepts of supervisory control of discrete-event systems modeled by deterministic finite automata with partial transition functions. For unexplained notions, the reader is referred to the monograph (Cassandras and Lafortune 2008).

Let \( \Sigma \) be a finite nonempty set whose elements are called events, and let \( \Sigma^* \) denote the set of all finite words (finite sequences of events) over \( \Sigma \); the empty word is denoted by \( \varepsilon \). Let \( |\Sigma| \) denote the cardinality of \( \Sigma \).

A generator is a quintuple \( G = (Q, \Sigma, f, q_0, Q_m) \), where \( Q \) is a finite nonempty set of states, \( \Sigma \) is a finite set of events (an event set), \( f : Q \times \Sigma \to Q \) is a partial transition function, \( q_0 \in Q \) is the initial state, and \( Q_m \subseteq Q \) is a set of marked states. In the usual way, the transition function \( f \) can be extended to the domain \( Q \times \Sigma^* \) by induction. The behavior of generator \( G \) is described in terms of languages. The language generated by \( G \) is the set \( L(G) = \{ s \in \Sigma^* | f(q_0, s) \in Q \} \), and the language marked by \( G \) is the set \( L_m(G) = \{ s \in \Sigma^* | f(q_0, s) \in Q_m \} \). Obviously, \( L_m(G) \subseteq L(G) \).

A (regular) language \( L \) over an event set \( \Sigma \) is a set \( L \subseteq \Sigma^* \) such that there exists a generator \( G \) with \( L_m(G) = L \). The prefix closure of a language \( L \) over \( \Sigma \) is the set \( \bar{L} = \{ w \in \Sigma^* | \text{there exists } u \in \Sigma^* \text{ such that } uu \in L \} \) of all prefixes of words of the language \( L \). A language \( L \) is prefix-closed if \( L = \bar{L} \).

A controlled generator over an event set \( \Sigma \) is a triple \( (G, \Sigma_c, \Gamma) \), where \( G \) is a generator over \( \Sigma \), \( \Sigma_c \subseteq \Sigma \) is a set of controllable events, \( \Sigma_u = \Sigma \setminus \Sigma_c \) is the set of uncontrollable events, and \( \Gamma = \{ \gamma \subseteq \Sigma | \Sigma_u \subseteq \gamma \} \) is the set of control patterns. A supervisor for the controlled generator \( (G, \Sigma_c, \Gamma) \) is a map \( S : L(G) \to \Gamma \). The closed-loop system associated with the controlled generator \( (G, \Sigma_c, \Gamma) \) and the supervisor \( S \) is defined as the minimal language \( L(S/G) \) such that the empty word \( \varepsilon \) belongs to \( L(S/G) \), and for any word \( s \) in \( L(S/G) \) such that \( sa \) is in \( L(G) \) and \( a \) in \( S(s) \), the word \( sa \) also belongs to \( L(S/G) \). We define the marked language of the closed-loop system as the intersection \( L_m(S/G) = L(S/G) \cap L_m(G) \). The intuition is that the supervisor disables some of the transitions of the generator \( G \), but it can never disable any transition under an uncontrollable event. If
the closed-loop system is nonblocking, which means that \( L_m(S/G) = L(S/G) \), then the supervisor \( S \) is called nonblocking.

Given a specification language \( K \) and a plant (generator) \( G \), the control objective of supervisory control is to find a nonblocking supervisor \( S \) such that \( L_m(S/G) = K \). For the monolithic case, such a supervisor exists if and only if the specification \( K \) is both controllable with respect to the plant language \( L(G) \) and uncontrollable event set \( \Sigma_u \), that is the inclusion \( K \Sigma_u \cap L \subseteq K \) is satisfied, and \( L_m(G) \)-closed, that is the equality \( K = K \cap L_m(G) \) is satisfied. For uncontrollable specifications, controllable sublanguages of the specification are considered instead. The notation \( \text{sup} C(K, L(G), \Sigma_u) \) denotes the supremal controllable sublanguage of the specification \( K \) with respect to the plant language \( L(G) \) and uncontrollable event set \( \Sigma_u \), which always exists and is equal to the union of all controllable sublanguages of the specification \( K \), see Wonham (2012).

A (natural) projection \( P : \Sigma^* \rightarrow \Sigma_0^* \), where \( \Sigma_0 \) is a subset of \( \Sigma \), is a homomorphism defined so that \( P(a) = \varepsilon \) for \( a \) in \( \Sigma \setminus \Sigma_0 \), and \( P(a) = a \) for \( a \) in \( \Sigma_0 \). The projection of a word is thus uniquely determined by projections of its letters. The inverse image of \( P \) is denoted by \( P^{-1} : \Sigma_0^* \rightarrow 2^{\Sigma^*} \). For three event sets \( \Sigma_i, \Sigma_j, \Sigma_\ell \), subsets of \( \Sigma \), we use the notation \( P_{i+j}^\ell \) to denote the projection from \( (\Sigma_i \cup \Sigma_j)^* \) to \( \Sigma_\ell^* \). If \( \Sigma_i \cup \Sigma_j = \Sigma \), we simplify the notation to \( P_\ell \). Similarly, the notation \( P_{i+k} \) stands for the projection from \( \Sigma^* \) to \( (\Sigma_i \cup \Sigma_k)^* \). The projection of a generator \( G \), denoted by \( P(G) \), is a generator whose behavior satisfies \( L(P(G)) = P(L(G)) \) and \( L_m(P(G)) = P(L_m(G)) \).

The synchronous product of languages \( L_1 \) over \( \Sigma_1 \) and \( L_2 \) over \( \Sigma_2 \) is defined as the language \( L_1 \parallel L_2 = P_1^{-1}(L_1) \cap P_2^{-1}(L_2) \), where \( P_i : (\Sigma_1 \cup \Sigma_2)^* \rightarrow \Sigma_i^* \) is a projection, for \( i = 1, 2 \). A similar definition for generators can be found in Cassandras and Lafortune (2008). The relation between the language definition and the generator definition is specified by the following equations. For generators \( G_1 \) and \( G_2 \), \( L(G_1 \parallel G_2) = L(G_1) \parallel L(G_2) \) and \( L_m(G_1 \parallel G_2) = L_m(G_1) \parallel L_m(G_2) \). In the automata framework, where a supervisor \( S \) has a finite representation as a generator, the closed-loop system is a synchronous product of the supervisor and the plant. Thus, we can write the closed-loop system as \( L(S/G) = L(S) \parallel L(G) \).

For a generator \( G \) over \( \Sigma \), let \( \Sigma_r(G) = \{ a \in \Sigma \mid \text{there are words } u,v \in \Sigma^* \text{ such that } uav \in L(G) \} \) denote the set of all events appearing in words of the language \( L(G) \). Generators \( G_1 \) and \( G_2 \) are conditionally independent with respect to a generator \( G_k \) if all events shared by the subsystems appear in the generator \( G_k \), that is, if the inclusion \( \Sigma_r(G_1) \cap \Sigma_r(G_2) \subseteq \Sigma_r(G_k) \) is satisfied. In other words, there is no simultaneous move in both generators \( G_1 \) and \( G_2 \) without the generator \( G_k \) being also involved.

Now, the notion of conditional decomposability is simplified compared to our previous work (Komenda et al. 2012c), but still equivalent.

**Definition 1** A language \( K \) is conditionally decomposable with respect to event sets \( \Sigma_1, \Sigma_2, \Sigma_k \), where \( \Sigma_1 \cap \Sigma_2 \subseteq \Sigma_k \subseteq \Sigma_1 \cup \Sigma_2 \), if \( K = P_{i+k}(K) \parallel P_{2+k}(K) \),

where \( P_{i+k} : (\Sigma_1 \cup \Sigma_2)^* \rightarrow (\Sigma_i \cup \Sigma_k)^* \) is a projection, for \( i = 1, 2 \).

Note that there always exists an extension of \( \Sigma_k \) which satisfies this condition; \( \Sigma_k = \Sigma_1 \cup \Sigma_2 \) is a trivial example. Here the index \( k \) is related to projection \( P_k \) used later in the paper. There exists a polynomial algorithm to check this condition, and to extend the event set to satisfy the condition, see Komenda et al. (2012b). However, the question which
extension is the most appropriate requires further investigation. In Section 3, we show that to find the minimal extension is NP-hard.

Languages $K$ and $L$ are \textit{synchronously nonconflicting} if $K \parallel L = K \parallel L$.

\textbf{Lemma 1} Let $K$ be a language. If the language $K$ is conditionally decomposable, then the languages $P_{1+k}(K)$ and $P_{2+k}(K)$ are synchronously nonconflicting.

\textit{Proof} Assume that the language $K$ is conditionally decomposable. From a simple observation that $K \subseteq P_{i+k}(P_{i+k}(K))$, for $i = 1, 2$, we immediately obtain that $K \subseteq P_{1+k}(K) \parallel P_{2+k}(K)$. As the prefix-closure is a monotone operation,

$$K \subseteq P_{1+k}(K) \parallel P_{2+k}(K) \subseteq P_{1+k}(K) \parallel P_{2+k}(K) = K,$$

which proves the lemma. \hfill \Box

The following example shows that there exists, in general, no relation between the conditional decomposability of languages $K$ and $\overline{K}$.

\textbf{Example 1} Let $\Sigma_1 = \{a_1, b_1, a, b\}$, $\Sigma_2 = \{a_2, b_2, a, b\}$, and $\Sigma_k = \{a, b\}$ be event sets, and define the language $K = \{a_1a_2a, a_2a_1a, b_1b_2b, b_2b_1b\}$. Then $P_{1+k}(K) = \{a_1a, b_1b\}$, $P_{2+k}(K) = \{a_2a, b_2b\}$, and $K = P_{1+k}(K) \parallel P_{2+k}(K)$. Notice that whereas $a_1b_2$ is in $P_{1+k}(K) \parallel P_{2+k}(K)$, $a_1b_2$ is not in $K$, which means that the language $K$ is not conditionally decomposable.

On the other hand, consider the language $L = \{\varepsilon, ab, ba, abc, bac\}$ over the event set \{$a, b, c\$} with $\Sigma_1 = \{a, c\}$, $\Sigma_2 = \{b, c\}$, $\Sigma_k = \{c\}$. Then $L = P_{1+k}(L) \parallel P_{2+k}(L) = P_{1+k}(L) \parallel P_{2+k}(L)$, and it is obvious that $L \neq \overline{L}$.

\section{3 Conditional decomposability minimal extension problem}

We have defined conditional decomposability only for two event sets, but the definition can be extended to more event sets as follows. A language $K$ is \textit{conditionally decomposable} with respect to event sets $(\Sigma_i)_{i=1}^n$, for some $n \geq 2$, and an event set $\Sigma_k$, where $\Sigma_k \subseteq \bigcup_{i=1}^n \Sigma_i$ contains all shared events, that is, it satisfies

$$\Sigma_s := \bigcup_{i \neq j}(\Sigma_i \cap \Sigma_j) \subseteq \Sigma_k,$$

if

$$K = \bigparallel_{i=1}^n P_{i+k}(K).$$

The conditional decomposability minimal extension problem is to find a minimal extension (with respect to set inclusion) of the event set $\Sigma_s$ of all shared events so that the language is conditionally decomposable with respect to given event sets and the extension of $\Sigma_s$. The optimization problem can be reformulated to a decision version as follows.

\textbf{Problem 1 (CD MIN EXTENSION)}

\textit{INSTANCE:} A language $K$ over an event set $\Sigma = \bigcup_{i=1}^n \Sigma_i$, where $n \geq 2$, and a positive integer $r \leq |\Sigma|$.

\textit{QUESTION:} Is the language $K$ conditionally decomposable with respect to event sets $(\Sigma_i)_{i=1}^n$ and $\Sigma_s \cup \Sigma_r$, where $|\Sigma_r| \leq r$?
We now prove that the CD MIN EXTENSION problem is NP-complete. This then immediately implies that the optimization problem of finding the minimal extension of the event set $\Sigma_s$ is NP-hard. On the other hand, it is not hard to see that the optimization problem is in PSPACE. Indeed, we can check all subsets generated one by one using the polynomial algorithm described in Komenda et al. (2012b).

To prove NP-completeness, we reduce the MINIMUM SET COVER problem to the CD MIN EXTENSION problem; the MINIMUM SET COVER problem is NP-complete (Garey and Johnson 1979).

**Problem 2 (MINIMUM SET COVER)**

INSTANCE: A collection $C$ of subsets of a finite set $S$, and a positive integer $t \leq |C|$. 

QUESTION: Does the collection $C$ contain a cover for the set $S$ of cardinality $t$ or less, that is, a subset $C'$ with $|C'| \leq t$ such that every element of the set $S$ belongs to at least one member of $C'$?

**Theorem 1** The CD MIN EXTENSION problem is NP-complete.

**Proof** First, we show that CD MIN EXTENSION is in NP. To do this, a Turing machine guesses a set $\Sigma_1$ of cardinality at most $r$ and uses Algorithm 1 of Komenda et al. (2012b) to verify in polynomial time whether the given language is conditionally decomposable with respect to the given event sets.

To prove the NP-hardness, consider an instance $(S, C)$ of the MINIMUM SET COVER problem as defined in Problem 2 such that the union of all elements of the collection $C$ covers the set $S$ (otherwise it is trivial to solve the problem). Denote $S = \{b_1, b_2, \ldots, b_n\}$ and $C = \{c_1, c_2, \ldots, c_m\}$.

We now construct a language $K$ over the event set $S \cup \{a_i \mid i = 1, 2, \ldots, n\} \cup C \cup \{a\}$ as follows. For each $b_i$ in $S$, let $C_{b_i} = \{c_j \mid b_i \in c_j\}$ be the set of all elements of the collection $C$ containing the element $b_i$. Then, for $C_{b_i} = \{c_{i_1}, c_{i_2}, \ldots, c_{i_{b_i}}\}$, where we assume without loss of generality that $i_1 < i_2 < \ldots < i_{b_i}$, add the two words $a_i a_{b_i}$ and $a_i c_{i_1} c_{i_2} \ldots c_{i_{b_i}} a$ to the language $K$. Then the language $K$ is

$$K = \sum_{i=1}^{n} \left( a_i a_{b_i} + a_i c_{i_1} c_{i_2} \ldots c_{i_{b_i}} a \right).$$

To demonstrate the construction, let $S = \{b_1, b_2, b_3, b_4, b_5\}$ and $C = \{c_1 = \{b_1, b_2, b_3\}, c_2 = \{b_2, b_3\}, c_3 = \{b_3, b_4\}, c_4 = \{b_4, b_5\}\}$. The generator for language $K$ is depicted in Fig. 1.

Note that $\{c_1, c_4\}$ is the minimum set cover. Next, we define two event sets

$$\Sigma_1 = S \cup \{a\} \cup \{a_i \mid i = 1, 2, \ldots, n\}$$

and

$$\Sigma_2 = C \cup \{a\} \cup \{a_i \mid i = 1, 2, \ldots, n\}.$$ 

As the intersection $S \cap C$ is empty, it gives that the event set $\Sigma_s = \{a\} \cup \{a_i \mid i = 1, 2, \ldots, n\}$. We now prove that there exists a minimum set cover of cardinality at most $r$ if and only if there exists an extension of the event set $\Sigma_s$ of cardinality at most $r$ making the language $K$ conditionally decomposable.

Assume that there exists a minimum set cover $C' = \{c_{i_1}, c_{i_2}, \ldots, c_{i_r}\} \subseteq C$ of cardinality $r$. We prove that the language $K$ is conditionally decomposable with respect to $\Sigma_1, \Sigma_2,$ and
Fig. 1 The generator for language $K$ corresponding to the MINIMUM SET COVER instance $(S, C)$, where $S = \{b_1, b_2, b_3, b_4, b_5\}$ and $C = \{c_1 = \{b_1, b_2, b_3\}, c_2 = \{b_2, b_4\}, c_3 = \{b_3, b_4\}, c_4 = \{b_4, b_5\}\}$

$\Sigma_k = \Sigma_s \cup \{c_{i_1}, c_{i_2}, \ldots, c_{i_r}\}$. The application of projection $P_{1+k}$ to language $K$ results in the language

$$P_{1+k}(K) = \sum_{i=1}^{n} \left( a_i a b_i + a_i P_{1+k} \left( c_{i_1} c_{i_2} \ldots c_{i_{b_i}} \right) a \right),$$

and the application of projection $P_{2+k}$ to language $K$ results in the language

$$P_{2+k}(K) = \sum_{i=1}^{n} \left( a_i a + a_i c_{i_1} c_{i_2} \ldots c_{i_{b_i}} a \right).$$

Note that the word $P_{1+k}(c_{i_1} c_{i_2} \ldots c_{i_{b_i}}) \in C'^{*}$ is nonempty because at least one set of the collection $C'$ covers the element $b_i$, for all $i = 1, 2, \ldots, n$. Let

$$X = C \setminus C'$$

denote the complement of the collection $C'$, then the intersection $X \cap S$ is empty. As it holds that $C_{b_i} \cap C \neq \emptyset$, for each element $b_i$ of the set $S$, the language $P_{1+k}^{-1} P_{1+k}(c_{i_1} c_{i_2} \ldots c_{i_{b_i}})$ is not a subset of the language $X^{*}$. It can be seen that the intersection $S^{*} c_{i_1} S^{*} c_{i_2} S^{*} \ldots S^{*} c_{i_{b_i}} S^{*} \cap X^{*} = \emptyset$ is empty, that the intersection $P_{1+k}^{-1} P_{1+k}(c_{i_1} c_{i_2} \ldots c_{i_{b_i}}) \cap S^{*} = \emptyset$ is empty, and that the intersection $P_{1+k}^{-1} P_{1+k}(c_{i_1}$
Then the parallel composition of both projections of the language $K$,

$$P_{1+k}(K) \parallel P_{2+k}(K)$$

$$= \sum_{i=1}^{n} (X^*a_iX^*aX^*b_iX^* + X^*a_iP_{1+k}(c_{i_1}c_{i_2} \ldots c_{i_k}))aX^*)$$

$$\cap \sum_{i=1}^{n} (S^*a_iS^*aS^* + S^*a_iS^*c_{i_1}S^*c_{i_2}S^* \ldots S^*c_{i_k}S^*aS^*)$$

$$= \sum_{i=1}^{n} (a_ia_i + a_i c_{i_1} c_{i_2} \ldots c_{i_k} a) = K,$$

is equal to $K$.

On the other hand, let $\Sigma_r \subseteq S \cup C$ be an extension of the event set $\Sigma_s$ of cardinality $r$ such that the language $K$ is conditionally decomposable with respect to event sets $\Sigma_1$, $\Sigma_2$, and $\Sigma_k = \Sigma_1 \cup \Sigma_r$. Consider a symbol $b_i$ and two corresponding words $a_iab_i$ and $a_ici_1c_{i_2} \ldots c_{i_k}a$ from the language $K$. If $\Sigma_r \cap \{b_i, c_{i_1}, c_{i_2}, \ldots, c_{i_k}\} = \emptyset$, then the projections of these words to event sets $\Sigma_2 \cup \Sigma_k$ and $\Sigma_1 \cup \Sigma_k$ are, respectively, $P_{2+k}(a_iab_i) = a_i a$ and $P_{1+k}(a_ici_1c_{i_2} \ldots c_{i_k}a) = a_i a$. But then the word $a_ici_1c_{i_2} \ldots c_{i_k}a \notin K$ belongs to $P_{1+k}(K) \parallel P_{2+k}(K)$, which is a contradiction. Hence, at least one of the symbols $b_i$, $c_{i_1}$, $c_{i_2}$, $\ldots$, $c_{i_k}$ must belong to the set $\Sigma_r$. In other words, at least one of these symbols covers the symbol $b_i$. We can now construct a covering $C' \subseteq C$ of cardinality at most $r$ as follows. For each $c$ in $\Sigma_r$, add the set $c$ to the covering $C'$, and for each $b$ in $\Sigma_r$, add any set $c$ from the set $C_b$ to the covering $C'$. It is then easy to see that the collection $C'$ covers the set $S$. 

Note that an immediate consequence of the construction is that the minimal extension problem is NP-hard even for finite languages and two event sets.

**Corollary 1** The minimal extension problem is NP-hard.

Similar minimal extension problems have been shown to be NP-hard in the literature, e.g., the minimal extension of observable event sets that guarantees observability of a language. However, unlike coobservability of decentralized control, conditional decomposability has an important property for large systems composed of many concurrent components—it can be checked in polynomial time in the number of components as shown in Komenda et al. (2012b). In addition, an algorithm is presented there to compute an extension (but not necessarily the minimal one) of the shared event set such that the language under consideration becomes conditionally decomposable with respect to the original event sets $\Sigma_1$ and $\Sigma_2$ and the new (coordinator) event set $\Sigma_k$.

## 4 Coordination control synthesis

In this section, we recall the coordination control problem and revise the necessary and sufficient conditions established in Komenda et al. (2011a, b, 2012c) under which the problem is solvable. This revision leads to a simplification of existing notions and proofs mentioned below.
We now summarize the results of this section compared to the existing results. The coordination control problem for non-prefix-closed languages was formulated in (Komenda et al. 2011a, Problem 7). The contribution of this paper is a simplification of the problem statement, namely, the prefix-closed part of the closed-loop system with a coordinator is shown to be a consequence of the non-prefix-closed case (see the note below the problem statement). The original definition of conditional controllability (Komenda et al. 2011a, Definition 9) is simplified in Definition 3. A simplified proof of Proposition 1 is presented (compare it with the proof of Komenda et al. (2011a, Proposition 10)). Proposition 2 is new. Theorem 4 is a simplified version of Theorem 18 stated in Komenda et al. (2011a) without proof.

Problem 3 (Coordination control problem) Consider generators $G_1$ and $G_2$ over $\Sigma_1$ and $\Sigma_2$, respectively, and a generator $G_k$ (called a coordinator) over $\Sigma_k$. Assume that generators $G_1$ and $G_2$ are conditionally independent with respect to coordinator $G_k$, and that a specification $K \subseteq L_m(G_1\|G_2\|G_k)$ and its prefix-closure $\overline{K}$ are conditionally decomposable with respect to event sets $\Sigma_1$, $\Sigma_2$, and $\Sigma_k$. The aim of the coordination control synthesis is to determine nonblocking supervisors $S_1$, $S_2$, and $S_k$ for respective generators such that

$$L_m(S_k/G_k) \subseteq P_k(K) \quad \text{and} \quad L_m(S_i/[G_i \| (S_k/G_k)]) \subseteq P_{i+k}(K), \ i = 1, 2,$$

and the closed-loop system with the coordinator satisfies

$$L_m(S_1/[G_1 \| (S_k/G_k)]) \parallel L_m(S_2/[G_2 \| (S_k/G_k)]) = K.$$

One could expect that the equality $L(S_1/[G_1 \| (S_k/G_k)]) \parallel L(S_2/[G_2 \| (S_k/G_k)]) = \overline{K}$ for prefix-closed languages should also be required in the statement of the problem. However, it is really sufficient to require only the equality for marked languages since it then implies that the equality $L(S_1/[G_1 \| (S_k/G_k)]) \parallel L(S_2/[G_2 \| (S_k/G_k)]) = \overline{K}$ holds true because

$$\overline{K} = L_m(S_1/[G_1 \| (S_k/G_k)]) \parallel L_m(S_2/[G_2 \| (S_k/G_k)])$$

$$\subseteq L_m(S_1/[G_1 \| (S_k/G_k)]) \parallel L_m(S_2/[G_2 \| (S_k/G_k)])$$

$$\subseteq P_{1+k}(K) \parallel P_{2+k}(K)$$

$$= \overline{K}.$$

Moreover, if such supervisors exist, their synchronous product is a nonblocking supervisor for the global plant, cf. (Komenda et al. 2011a).

Note that several conditions are required in the statement of the problem, namely, (i) the generators are conditionally independent with respect to the coordinator and (ii) the specification and its prefix-closure are conditionally decomposable with respect to event sets $\Sigma_1$, $\Sigma_2$, and $\Sigma_k$. These conditions can easily be fulfilled by the choice of an appropriate coordinator event set $\Sigma_k$. The reader is referred to Komenda et al. (2012b) for a polynomial algorithm extending a given event set so that the language becomes conditionally decomposable.

In the statement of the problem, we have mentioned the notion of a coordinator. The fundamental question is the construction of such a coordinator. We now discuss one of the possible constructions of a suitable coordinator, which has already been discussed in the literature Komenda et al. (2011a, b, 2012c). We recall it here for the completeness.
Algorithm 2 (Construction of a coordinator) Consider generators $G_1$ and $G_2$ over $\Sigma_1$ and $\Sigma_2$, respectively, and let $K$ be a specification. Construct an event set $\Sigma_k$ and a coordinator $G_k$ as follows:

1. Set $\Sigma_k = \Sigma_1 \cap \Sigma_2$ to be the set of all shared events.
2. Extend $\Sigma_k$ so that $K$ and $\overline{K}$ are conditional decomposable, for instance using a method described in Komenda et al. (2012b).
3. Let the coordinator $G_k = P_k(G_1) \parallel P_k(G_2)$.

So far, the only known condition ensuring that the projected generator is smaller than the original one is the observer property. Therefore, we might need to add step (2b) to extend the event set $\Sigma_k$ so that the projection $P_k$ is an $L(G_i)$-observer, for $i = 1, 2$, cf. Definition 2 below.

Note that if we generalize this approach to more than two subsystems, the set $\Sigma_k$ of step 1 is replaced with the set $\Sigma_s$ of all shared events defined in Section 3 above.

Definition 2 (Observer) The projection $P_k : \Sigma^* \rightarrow \Sigma_k^*$, where $\Sigma_k$ is a subset of $\Sigma$, is an $L$-observer for a language $L$ over $\Sigma$ if, for all words $t$ in $P_k(L)$ and $s$ in $\overline{L}$, the word $P_k(s)$ is a prefix of $t$ implies that there exists a word $u$ in $\Sigma^*$ such that $su$ is in $L$ and $P_k(su) = t$.

For a generator $G$ with $n$ states, the time and space complexity of the verification whether a projection $P$ is an $L(G)$-observer is $O(n^2)$, see Pena et al. (2008) and Bravo et al. (2012). An algorithm extending the event set to satisfy the property runs in time $O(n^3)$ and linear space. The most significant consequence of the observer property is the following theorem.

Theorem 3 (Wong 1998) If a projection $P$ is an $L(G)$-observer, for a generator $G$, then the minimal generator for the language $P(L(G))$ has no more states than the generator $G$.

This is an important result because it guarantees that the coordinator computed in Algorithm 2 is smaller than the plant whenever the projection $P_k$ is an $L(G_1) \parallel L(G_2)$-observer.

4.1 Conditional controllability

The concept of conditional controllability introduced in Komenda and van Schuppen (2008) and later studied in Komenda et al. (2011a, b, 2012c) plays the central role in the coordination control approach. In this paper, we revise and simplify this notion. In what follows, we use the notation $\Sigma_{i,u} = \Sigma_i \cap \Sigma_u$ to denote the set of locally uncontrollable events of the event set $\Sigma_i$.

Definition 3 Let $G_1$ and $G_2$ be generators over $\Sigma_1$ and $\Sigma_2$, respectively, and let $G_k$ be a coordinator over $\Sigma_k$. A language $K \subseteq L(G_1 \parallel G_2 \parallel G_k)$ is conditionally controllable for generators $G_1, G_2, G_k$ and uncontrollable event sets $\Sigma_{1,u}, \Sigma_{2,u}, \Sigma_{k,u}$ if

1. $P_k(K)$ is controllable with respect to $L(G_k)$ and $\Sigma_{k,u}$,
2. $P_{1+k}(K)$ is controllable with respect to $L(G_1) \parallel P_k(K)$ and $\Sigma_{1+k,u}$,
3. $P_{2+k}(K)$ is controllable with respect to $L(G_2) \parallel P_k(K)$ and $\Sigma_{2+k,u}$,

where $\Sigma_{i+k,u} = (\Sigma_i \cup \Sigma_k) \cap \Sigma_u$, for $i = 1, 2$. 

Springer
The difference between Definition 3 and the definition in previous papers is that in item 2 we write \( L(G_1) \parallel P_k(K) \) instead of \( L(G_1) \parallel P_k(K) \parallel P_{k+2}^k(L(G_2)) \parallel P_k(K) \). This is possible because the assumption \( K \subseteq L(G_1) \parallel G_2 \parallel G_k \) implies that \( P_k(K) \subseteq (P_{k\cap 2})^{-1} P_{k\cap 2}(L(G_2)) \), which results in the equality

\[
\overline{P_k(K)} \parallel P_{k+2}^k(L(G_2)) \parallel \overline{P_k(K)} = \overline{P_k(K)} \parallel P_{k\cap 2}^2(L(G_2)) \\
= \overline{P_k(K)} \cap (P_{k\cap 2})^{-1} P_{k\cap 2}^2(L(G_2)) \\
= \overline{P_k(K)}
\]

by Lemma 9 (see the Appendix). Hence we have the following.

**Lemma 2** Definition 3 and Komenda et al. (2011a, Definition 9) of conditional controllability are equivalent.

The following proposition demonstrates that every conditionally controllable and conditionally decomposable language is controllable.

**Proposition 1** Let \( G_i \) be a generator over \( \Sigma_i \), for \( i = 1, 2, k \), and let \( G = G_1 \parallel G_2 \parallel G_k \). Let \( K \subseteq L_m(G) \) be such a specification that the language \( \overline{K} \) is conditionally decomposable with respect to event sets \( \Sigma_1, \Sigma_2, \Sigma_k \), and conditionally controllable for generators \( G_1, G_2, G_k \) and uncontrollable event sets \( \Sigma_1, \Sigma_2, \Sigma_k \). Then the language \( K \) is controllable with respect to the plant language \( L(G) \) and uncontrollable event set \( \Sigma_u = \Sigma_{1, u} \cup \Sigma_{2, u} \).

**Proof** Since the language \( P_{1+k}(K) \) is controllable with respect to \( L(G_1) \parallel P_k(K) \) and \( \Sigma_{1+k, u} \), and \( P_{2+k}(K) \) is controllable with respect to \( L(G_2) \parallel P_k(K) \) and \( \Sigma_{2+k, u} \), Lemma 7 implies that the language \( \overline{K} = P_{1+k}(K) \parallel P_{2+k}(K) \) is controllable with respect to \( L(G_1) \parallel P_k(K) \parallel L(G_2) \parallel P_k(K) = L(G) \parallel P_k(K) \) and \( \Sigma_u \), where the equality is by commutativity of the synchronous product and by the fact that \( P_k(K) \subseteq L(G_k) \). As the language \( P_k(K) \) is controllable with respect to \( L(G_k) \) and \( \Sigma_{k,u} \), by Definition 3, the language \( L(G) \parallel P_k(K) \) is controllable with respect to \( L(G) \parallel L(G_k) = L(G) \) by Lemma 7. Finally, by Lemma 8, \( \overline{K} \) is controllable with respect to \( L(G) \) and \( \Sigma_u \), which means that \( K \) is controllable with respect to \( L(G) \) and \( \Sigma_u \).

On the other hand, controllability does not imply conditional controllability.

**Example 2** Let \( G \) be a generator such that \( L(G) = \{au\} \parallel \{bu\} = \{abu, ba\} \). Then the language \( K = \{a\} \) is controllable with respect to \( L(G) \) and \( \Sigma_u = \{a\} \). Moreover, both languages \( K \) and \( \overline{K} \) are conditionally decomposable with respect to event sets \( \{a, u\} \), \( \{b, u\} \), and \( \Sigma_k = \{u\} \), but the language \( P_k(K) = \{e\} \) is not controllable with respect to \( L(G_k) = P_k(L(G)) = \{u\} \) and \( \Sigma_{k,u} = \{u\} \).

However, we show below that if the observer property and local control consistency (LCC) are satisfied, the previous implication holds. To prove this, we need the following definition of LCC. Note that unlike our previous papers, we use a weaker notion of local control consistency (LCC) presented in Schmidt and Breindl (2011) instead of output control consistency (OCC).

**Definition 4** (LCC) Let \( L \) be a prefix-closed language over \( \Sigma \), and let \( \Sigma_0 \) be a subset of \( \Sigma \). The projection \( P_0 : \Sigma^* \rightarrow \Sigma_0^* \) is locally control consistent (LCC) with respect to a
word \( s \in L \) if for all events \( \sigma_u \in \Sigma_0 \cap \Sigma_u \) such that \( P_0(s)\sigma_u \in P_0(L) \), it holds that either there does not exist any word \( u \in (\Sigma \setminus \Sigma_0)^* \) such that \( su_\sigma u \in L \), or there exists a word \( u \in (\Sigma_u \setminus \Sigma_0)^* \) such that \( su_\sigma u \in L \). The projection \( P_0 \) is LCC with respect to a language \( L \) if \( P_0 \) is LCC for all words of \( L \).

Now the opposite implication to the one proven in Proposition 1 can be stated.

**Proposition 2** Let \( L \) be a prefix-closed language over \( \Sigma \), and let \( K \subseteq L \) be a language that is controllable with respect to \( L \) and \( \Sigma_u \). If, for \( i \in \{ k, 1+k, 2+k \} \), the projection \( P_i \) is an \( L \)-observer and LCC for \( L \), then the language \( K \) is conditionally controllable.

**Proof** Let \( s \in \overline{P}_k(K) \), \( a \in \Sigma_{k,u} \), and \( sa \in P_k(L) \). Then there exists a word \( w \) in \( K \) such that \( P_k(w) = s \). By the observer property, there exists a word \( u \) in \( (\Sigma \setminus \Sigma_k)^* \) such that \( wua \in L \) and \( P_k(wua) = sa \). By LCC, there exists another word \( u' \) in \( (\Sigma_u \setminus \Sigma_k)^* \) such that \( wu'a \in L \), that is, \( wu'a \) is in \( K \) by controllability. Hence, \( sa \in \overline{P}_k(K) \).

Let \( s \in \overline{P}_{1+k}(K) \), \( a \in \Sigma_{1+k,u} \), and \( sa \in L(G_1) \parallel \overline{P}_k(K) \). Then there exists a word \( w \) in \( K \) such that \( P_{1+k}(w) = s \). By the observer property, there exists a word \( u \) in \( (\Sigma \setminus \Sigma_{1+k})^* \) such that \( wua \in L \) and \( P_{1+k}(wua) = sa \). By LCC, there exists another word \( u' \) in \( (\Sigma_u \setminus \Sigma_{1+k})^* \) such that \( wu'a \in L \), that is, \( wu'a \) is in \( K \) by controllability. Hence, \( sa \in \overline{P}_{1+k}(K) \).

The proof for the case of \( k + 2 \) is similar to that of \( k + 1 \). \( \Box \)

### 4.2 Conditionally closed languages

In this subsection we turn our attention to general specification languages that need not be prefix-closed. Analogously to the notion of \( L_m(G) \)-closed languages, we recall the notion of conditionally-closed languages defined in Komenda et al. (2011a).

**Definition 5** A nonempty language \( K \) over \( \Sigma \) is **conditionally closed** for generators \( G_1, G_2, G_k \) if

1. \( P_k(K) \) is \( L_m(G_k) \)-closed,
2. \( P_{1+k}(K) \) is \( L_m(G_1) \parallel P_k(K) \)-closed,
3. \( P_{2+k}(K) \) is \( L_m(G_2) \parallel P_k(K) \)-closed.

If a language \( K \) is conditionally closed and conditionally controllable, then there exists a nonblocking supervisor \( S_k \) such that \( L_m(S_k/G_k) = P_k(K) \), which follows from the basic theorem of supervisory control applied to languages \( P_k(K) \) and \( L(G_k) \), see Cassandras and Lafortune (2008).

As noted in (Cassandras and Lafortune 2008, page 164), if \( K \subseteq L_m(G) \) is \( L_m(G) \)-closed, then so is the supremal controllable sublanguage of \( K \). However, this does not imply that the language \( P_k(K) \) is \( L_m(G_k) \)-closed, for any generator \( G = G_1 \parallel G_2 \parallel G_k \) such that the coordinator \( G_k \) makes generators \( G_1 \) and \( G_2 \) conditionally independent.

**Example 3** Let the event sets be \( \Sigma_1 = \{a_1, a\} \), \( \Sigma_2 = \{a_2, a\} \), and \( \Sigma_k = \{a\} \), respectively, and let the specification language be \( K = \{a_1 a_2 a, a_2 a_1 a\} \). Then the application of projections results in languages \( P_{1+k}(K) = \{a_1 a\} \), \( P_{2+k}(K) = \{a_2 a\} \), and \( P_k(K) = \{a\} \), and the language \( K = P_{1+k}(K) \parallel P_{2+k}(K) \) is conditionally decomposable. Define generators \( G_1, G_2, G_k \) so that \( L_m(G_1) = P_{1+k}(K) \), \( L_m(G_2) = P_{2+k}(K) \), and \( L_m(G_k) = \overline{P}_k(K) = \{e, a\} \).
Then $L_m(G) = K$ and the language $K$ is $L_m(G)$-closed. However, the language $P_k(K) \subset P_k(K)$ is not $L_m(G_k)$-closed.

4.3 Existence of supervisors

The following theorem is a revised version (based on the simplification of conditional controllability, Definition 3) of a result presented without proof in Komenda et al. (2011a).

**Theorem 3** Consider the setting of Problem 3. There exist nonblocking supervisors $S_1$, $S_2$, $S_k$ such that

$$L_m(S_1/[G_1 \parallel (S_k/G_k)]) \parallel L_m(S_2/[G_2 \parallel (S_k/G_k)]) = K$$

if and only if the specification language $K$ is both conditionally controllable with respect to generators $G_1$, $G_2$, $G_k$ and uncontrollable event sets $\Sigma_{1,u}$, $\Sigma_{2,u}$, $\Sigma_{k,u}$, and conditionally closed with respect to generators $G_1$, $G_2$, $G_k$.

**Proof** Let $K$ satisfy the assumptions, and let $G = G_1\|G_2\|G_k$ be the global plant. As the language $K$ is a subset of $L_m(G)$, its projection $P_k(K)$ is a subset of $L_m(G_k)$. By the assumption, the language $P_k(K)$ is $L_m(G_k)$-closed and controllable with respect to $L(G_k)$ and $\Sigma_{k,u}$. By the basic theorem of supervisory control (Ramadge and Wonham 1987) there exists a nonblocking supervisor $S_k$ such that $L_m(S_k/G_k) = P_k(K)$. As the language $P_{1+k}(K)$ is a subset of languages $L_m(G_1\|G_k)$ and $(P_{k+1}^{-1}) P_k(K)$, we have that $P_{1+k}(K)$ is included in $L_m(G_1) \parallel P_k(K)$. These relations and the assumption that the system is conditionally controllable and conditionally closed imply the existence of a nonblocking supervisor $S_1$ such that $L_m(S_1/[G_1 \parallel (S_k/G_k)]) = P_{1+k}(K)$. A similar argument shows that there exists a nonblocking supervisor $S_2$ such that $L_m(S_2/[G_2 \parallel (S_k/G_k)]) = P_{2+k}(K)$. Since $K$ and $\overline{K}$ are conditionally decomposable, it follows that $L_m(S_1/[G_1 \parallel (S_k/G_k)]) \parallel L_m(S_2/[G_2 \parallel (S_k/G_k)]) = P_{1+k}(K) \parallel P_{2+k}(K) = K$.

To prove the converse implication, the projections $P_k$, $P_{1+k}$, $P_{2+k}$ are applied to Eq. 1, which can be rewritten as $K = L_m(S_1/[G_1 \parallel S_2\|G_2 \parallel S_k\|G_k])$. Thus, the projection $P_k(K) = P_k (L_m(S_1/[G_1 \parallel S_2\|G_2 \parallel S_k\|G_k]))$ is a subset of $L_m(S_k\|G_k) = L_m(S_k/G_k)$. On the other hand, $L_m(S_k/G_k) \subseteq P_k(K)$, cf. Problem 3. Hence, by the basic controllability theorem, the language $P_k(K)$ is both controllable with respect to $L(G_k)$ and $\Sigma_{k,u}$ and $L_m(G_k)$-closed. As $\Sigma_{1+k} \cap \Sigma_{2+k} = \Sigma_k$, the application of projection $P_{1+k}$ to Eq. 1 and assumptions of Problem 3 give that $P_{1+k}(K) \subseteq L_m(S_1/[G_1 \parallel (S_k/G_k)]) \subseteq P_{1+k}(K)$. Taking $G_1\|S_k/G_k$ as a new plant, we get from the basic supervisory control theorem that the language $P_{1+k}(K)$ is controllable with respect to $L(G_1\|S_k/G_k)$ and $\Sigma_{1+k,u}$, and that it is $L_m(G_1\|S_k/G_k)$-closed. The case of the language $P_{2+k}(K)$ is analogous.

5 Supremal conditionally controllable sublanguages

Necessary and sufficient conditions for the existence of nonblocking supervisors $S_1$, $S_2$, and $S_k$ that achieve a considered specification language using our coordination control architecture have been presented in Theorem 4. However, in many cases control specifications fail to be conditionally controllable and, similarly as in the monolithic supervisory control, supremal conditionally controllable sublanguages should be investigated.

Let $\text{supc}(K, L, (\Sigma_{1,u}, \Sigma_{2,u}, \Sigma_{k,u}))$ denote the supremal conditionally controllable sublanguage of $K$ with respect to $L = L(G_1\|G_2\|G_k)$ and sets of uncontrollable events
\( \Sigma_{1,u}, \Sigma_{2,u}, \Sigma_{k,u} \). The supremal conditionally controllable sublanguage always exists, cf. (Komenda et al. 2011b) for the case of prefix-closed languages.

**Theorem 4** The supremal conditionally controllable sublanguage of a given language \( K \) always exists and is equal to the union of all conditionally controllable sublanguages of the language \( K \).

**Proof** Let \( I \) be an index set, and let \( K_i \), for \( i \in I \), be conditionally controllable sublanguages of \( K \subseteq L(G_1 \| G_2 \| G_k) \). To prove that the language \( P_k(\cup_{i \in I} K_i) \) is controllable with respect to \( L(G_k) \) and \( \Sigma_{k,u} \), note that

\[
P_k(\cup_{i \in I} K_i) \Sigma_{k,u} \cap L(G_k) = \cup_{i \in I} \left( P_k(K_i) \Sigma_{k,u} \cap L(G_k) \right) \\
\subseteq \cup_{i \in I} P_k(K_i) \\
= P_k(\cup_{i \in I} K_i),
\]

where the inclusion is by controllability of the language \( P_k(K_i) \) with respect to \( L(G_k) \) and \( \Sigma_{k,u} \). Next, to prove that

\[
P_{1+k}(\cup_{i \in I} K_i) \Sigma_{1+k,u} \cap L(G_1) \| P_k(\cup_{i \in I} K_i) \subseteq P_{1+k}(\cup_{i \in I} K_i),
\]

note that

\[
P_{1+k}(\cup_{i \in I} K_i) \Sigma_{1+k,u} \cap L(G_1) \| P_k(\cup_{i \in I} K_i) \\
= \cup_{i \in I} \left( P_{1+k}(K_i) \Sigma_{1+k,u} \cap L(G_1) \| P_k(K_i) \right) \\
= \cup_{i \in I} \cup_{j \in I} \left( P_{1+k}(K_i) \Sigma_{1+k,u} \cap L(G_1) \| P_k(K_j) \right).
\]

Consider two different indexes \( i \) and \( j \) from \( I \) such that

\[
P_{1+k}(K_i) \Sigma_{1+k,u} \cap L(G_1) \| P_k(K_j) \not\subseteq P_{1+k}(\cup_{i \in I} K_i).
\]

Then there exist a word \( x \) in \( P_{1+k}(K_i) \) and an uncontrollable event \( u \) in \( \Sigma_{1+k,u} \) such that \( xu \) belongs to the language \( L(G_1) \| P_k(K_j) \), and \( xu \) does not belong to \( P_{1+k}(\cup_{i \in I} K_i) \). It follows that \( P_k(x) \) belongs to \( P_k(K_i) \) and \( P_k(xu) \) belongs to \( P_k(K_j) \). If \( P_k(xu) \) belongs to \( P_k(K_i) \), then \( xu \) belongs to \( L(G_1) \| P_k(K_i) \), and controllability of the language \( P_{1+k}(K_i) \) with respect to \( L(G_1) \| P_k(K_i) \) implies that \( xu \) belongs to \( P_{1+k}(\cup_{i \in I} K_i) \); hence, \( P_k(xu) \) does not belong to \( P_k(K_i) \). If the event \( u \) does not belong to \( \Sigma_{k,u} \), then \( P_k(xu) = P_k(x) \) belongs to \( P_k(K_j) \), which is not the case. Thus, \( u \) belongs to \( \Sigma_{k,u} \). As \( P_k(K_i) \cup P_k(K_j) \) is a subset of \( L(G_k) \), we get that \( P_k(xu) = P_k(x)u \) belongs to \( L(G_k) \). However, controllability of the language \( P_k(K_i) \) with respect to \( L(G_k) \) and \( \Sigma_{k,u} \) implies that the word \( P_k(xu) \) belongs to \( P_k(K_i) \). This is a contradiction.

As the case for the projection \( P_{2+k} \) is analogous, the proof is complete. \( \square \)

Still, it is a difficult problem to compute a supremal conditional controllable sublanguage. Consider the setting of Problem 3 and define the languages

\[
\sup C_k = \sup C(P_k(K), L(G_k), \Sigma_{k,u}) \\
\sup C_{1+k} = \sup C(P_{1+k}(K), L(G_1) \| \sup C_k, \Sigma_{1+k,u}) \quad (*) \\
\sup C_{2+k} = \sup C(P_{2+k}(K), L(G_2) \| \sup C_k, \Sigma_{2+k,u})
\]

Interestingly, the following inclusion always holds. \( \square \)
Lemma 3 Consider the setting of Problem 3, and languages defined in (\*). Then the language \( P_k(\sup C_{i+k}) \) is a subset of the language \( \sup C_i \), for \( i = 1, 2 \).

Proof By definition, the language \( P_k(\sup C_{i+k}) \) is a subset of languages \( \sup C_{i+k} \) and \( P_k(K) \). To prove that \( P_k(\sup C_{i+k}) \) is a subset of \( \sup C_i \), we prove that the language \( \sup C_{i+k} \cap P_k(K) \) is a subset of \( \sup C_i \). To do this, it is sufficient to show that the language \( \sup C_{i+k} \cap P_k(K) \) is controllable with respect to \( L(G_k) \) and \( \Sigma_{k,u} \).

Thus, consider a word \( s \) in \( \sup C_{i+k} \cap P_k(K) \), an uncontrollable event \( u \) in \( \Sigma_{k,u} \), and the word \( su \) in \( L(G_k) \). By controllability of \( \sup C_i \), the word \( su \) belongs to \( \sup C_i \), which is a subset of \( P_k(K) \). That is, there exists a word \( v \) such that \( suv \) is in \( \sup C_i \), which is a subset of \( P_k(K) \). This means that the word \( suv \) belongs to \( \sup C_i \cap P_k(K) \), which implies that the word \( su \) is in \( \sup C_{i+k} \cap P_k(K) \). This completes the proof.

It turns out that if the converse inclusion also holds, then we immediately obtain the supremal conditionally-controllable sublanguage.

Theorem 5 Consider the setting of Problem 3, and languages defined in (\*). If \( \sup C_i \) is a subset of \( P_k(\sup C_{i+k}) \), for \( i = 1, 2 \), then

\[
\sup C_{i+k} \parallel \sup C_{2+k} = \sup C(K, L, (\Sigma_{1,u}, \Sigma_{2,u}, \Sigma_{k,u})) .
\]

Proof Let \( \sup C = \sup C(K, L, (\Sigma_{1,u}, \Sigma_{2,u}, \Sigma_{k,u})) \) and \( M = \sup C_{1+k} \parallel \sup C_{2+k} \). To prove that \( M \) is a subset of \( \sup C \), we show that (i) \( M \) is a subset of \( K \) and (ii) \( M \) is conditionally controllable with respect to generators \( G_1, G_2, G_k \) and uncontrollable event sets \( \Sigma_{1,u}, \Sigma_{2,u}, \Sigma_{k,u} \). To this aim, notice that \( M \) is a subset of \( P_{1+k}(K) \parallel P_{2+k}(K) = K \), because \( K \) is conditionally decomposable. Moreover, by Lemmas 9 and 3, the language \( P_k(M) = P_k(\sup C_{1+k}) \cap P_k(\sup C_{2+k}) = \sup C_k \), which is controllable with respect to \( L(G_k) \) and \( \Sigma_{k,u} \). Similarly, \( P_{i+k}(M) = \sup C_{i+k} \parallel P_k(\sup C_{j+k}) = \sup C_{i+k} \parallel \sup C_j = \sup C_{i+k} \), for \( j \neq i \), which is controllable with respect to \( L(G_1) \parallel P_k(M) \). Hence, \( M \) is a subset of \( \sup C \).

To prove the opposite inclusion, it is sufficient, by Lemma 10, to show that the language \( P_{1+k}(\sup C) \) is a subset of \( \sup C_{i+k} \), for \( i = 1, 2 \). To prove this note that the language \( P_{1+k}(\sup C) \) is controllable with respect to \( L(G_1) \parallel P_k(\sup C) \) and \( \Sigma_{1+k,u} \), and the language \( L(G_1) \parallel P_k(\sup C) \) is controllable with respect to \( L(G_1) \parallel \sup C_k \) and \( \Sigma_{1+k,u} \), by Lemma 7, because the language \( P_k(\sup C) \) being controllable with respect to \( L(G_k) \) implies that it is also controllable with respect to \( \sup C_k \), which is a subset of \( L(G_1) \parallel \sup C_k \). By Lemma 8, the language \( P_{1+k}(\sup C) \) is controllable with respect to \( L(G_1) \parallel \sup C_k \) and \( \Sigma_{1+k,u} \), which implies that \( P_{1+k}(\sup C) \) is a subset of \( \sup C_{1+k} \). The other case is analogous. Hence, the language \( \sup C \) is a subset of \( M \) and the proof is complete.

Example 4 This example demonstrates that the language \( \sup C_k \) is not always included in the language \( P_k(\sup C_{i+k}) \). Moreover, it does not hold even if projections are observers or satisfy the LCC property.

Consider systems \( G_1 \) and \( G_2 \) shown in Fig. 2, and the specification \( K \) as shown in Fig. 3. Controllable events are \( \Sigma_c = \{a_1, a_2, c\} \), and coordinator events are \( \Sigma_k = \{a_1, a_2, c, u\} \). Construct the coordinator \( G_k = P_k(G_1) \parallel P_k(G_2) \). It can be verified that \( K \) is conditionally decomposable, \( \sup C_k = \{a_1a_2, a_2a_1, \} \), \( \sup C_{1+k} = \{a_2a_1u_1\} \), and \( \sup C_{2+k} = \{a_1a_2u_2\} \).

Hence, \( \sup C_k \) is not a subset of \( P_k(\sup C_{i+k}) \).
It can be verified that projections $P_k$, $P_{1+k}$, $P_{2+k}$ are $L(G_1\parallel G_2)$-observers and LCC for the language $L(G_1\parallel G_2)$. ▷

Recall that it is still open how to compute the supremal conditionally-controllable sub-language for a general, non-prefix-closed language. Consider the example above and note that the words $a_1a_2$ and $a_2a_1$ from $\text{sup} C_k$ do not appear in the projection of the supremal conditionally-controllable sublanguage, that is, no words with both letters $a_1$ and $a_2$ appear in the supremal conditionally-controllable sublanguage. Thus, we can remove these words from $\text{sup} C_k$ (basically from the coordinator) and recompute the supremal controllable sublanguage (denoted by $\text{sup} C'_k$), i.e., $\text{sup} C'_k = \text{sup} C(\cap_{i=1,2} P_k(\text{sup} C_{i+k}), L(G_k), \Sigma_{k,u}) = \{\varepsilon\}$ and, similarly, recompute $\text{sup} C_{i+k}$ using $\text{sup} C'_k$ instead of $\text{sup} C_k$. Note that the plant is changed because the coordinator restricts it more than before. An application of Theorem 6 could thus be as follows. If $\text{sup} C_k \not\subseteq P_k(\text{sup} C_{i+k})$, then the natural approach seems to be to remove from $\text{sup} C_k$ all words violating the inclusion, and to recompute $\text{sup} C_{i+k}$, for $i = 1, 2$, with respect to this new $\text{sup} C'_k$, that is

\[
\begin{align*}
\text{sup} C_k' &= \text{sup} C(P_k(\text{sup} C_{1+k}) \cap P_k(\text{sup} C_{2+k}), L(G_k), \Sigma_{k,u}) \\
\text{sup} C'_{1+k} &= \text{sup} C(\text{sup} C_{1+k}, L(G_1) \parallel \text{sup} C_k', \Sigma_{1+k,u}) \\
\text{sup} C'_{2+k} &= \text{sup} C(\text{sup} C_{2+k}, L(G_2) \parallel \text{sup} C_k', \Sigma_{2+k,u})
\end{align*}
\]

(\*)

In our example, we get that $\text{sup} C'_{1+k} = \{\varepsilon\}$ and $\text{sup} C'_{2+k} = \{\varepsilon\}$ satisfy the assumption that $\text{sup} C'_k \subseteq P_k(\text{sup} C'_{i+k})$, for $i = 1, 2$, hence Theorem 6 applies. It is not yet clear whether this method can be used in general, namely whether it always terminates and the result is the supremal conditionally-controllable sublanguage. It is only known that if it terminates, the result is conditionally controllable (see the end of Section 6 for more discussion). Another problem is that it requires to compute the projection, which can be exponential in general, because the observer property is not ensured. One of the natural investigations of this problem is to work with nondeterministic representations. Several attempts in this direction were done in the literature although they usually handle the case where only the plant is non-deterministic, while the specification is deterministic, see, e.g., Su et al. (2010, 2012). Even more, it is a question how to test the inclusion from Theorem 6.

Fig. 2 Generators $G_1$ and $G_2$

Fig. 3 Specification $K$
Finally, if \( \text{sup } C_{i+k} \) and \( \text{sup } C'_k \) are nonconflicting, the language \( \text{sup } C_{i+k} \| \text{sup } C'_k \) is controllable with respect to \( L(G_i) \| \text{sup } C'_k \). By Lemma 7. This observation gives the following result for prefix-closed languages.

**Lemma 4** Let \( K = \overline{K} \subseteq L = L(G_1)\|L(G_2)\|L(G_k) \), where \( G_i \) is a generator over \( \Sigma_i \), for \( i = 1, 2, k \). Assume that \( K \) is conditionally decomposable, and define the languages \( \text{sup } C_{i+k} \) and \( \text{sup } C_{2+k} \) in (*) . If \( \text{sup } C_k \not\subseteq P_k(\text{sup } C_{i+k}) \), for \( i \in \{1, 2\} \), define the language \( \text{sup } C'_k \) as in (**). Then the language

\[
\text{sup } C_{i+k} \| \text{sup } C_{2+k} \| \text{sup } C'_k
\]

is conditionally controllable with respect to \( G_1, G_2, G_k \) and \( \Sigma_{1,u}, \Sigma_{2,u}, \Sigma_{k,u} \).

Note that if we have any specification \( K \), which is conditionally decomposable, then the specification \( K \| L \) is also conditionally decomposable. The opposite is not true.

**Lemma 5** Let \( K \) be conditionally decomposable with respect to event sets \( \Sigma_1, \Sigma_2, \Sigma_k \), and let \( L = L_1 \| L_2 \| L_k \), where \( L_i \) is over \( \Sigma_i \), for \( i = 1, 2, k \). Then the language \( K \| L \) is conditionally decomposable with respect to event sets \( \Sigma_1, \Sigma_2, \Sigma_k \).

**Proof** By the assumption we have that \( K = P_{1+k}(K)\|P_{2+k}(K) \). Then

\[
K \| L = P_{1+k}(K)\|P_{2+k}(K)\|L_1\|L_2\|L_k = P_{1+k}(K)\|L_1\|L_k \| P_{2+k}(K)\|L_2\|L_k
= P_{1+k}(K)\|L_1\|L_k \| P_{2+k}(K)\|L_2\|L_k
\]

where the last equality is by Lemma 9. By Lemma 12, \( K \| L \) is conditionally decomposable with respect to event sets \( \Sigma_1, \Sigma_2, \Sigma_k \).

**Example 5** Consider a situation at a railway station. There are several tracks that cross each other at some points. Obviously, the traffic has to be controlled at those points. For simplicity, we consider only two one-way tracks that cross at some point, that is, trains going from west to east use track one, while trains going from east to west use track two. The traffic is controlled by traffic lights.

Thus, consider the railway crossroad with two traffic lights, \( S_1 \) and \( S_2 \), and two entry points \( x_1, x_3 \) and two exit points \( x_2, x_4 \), as depicted in Fig. 4. Each traffic light has values \( g_i \) (green) and \( r_i \) (red), for \( i = 1, 2 \). Colors of the traffic lights are controllable. The plant is then given as a parallel composition of two systems \( G_1 \) and \( G_2 \) depicted in Fig. 5. For safety reasons, each system is able to set the traffic light to red at any moment. It can set the traffic light to green and the trains are detected entering \( (x_1 \text{ or } x_3) \) and leaving \( (x_2 \text{ or } x_4) \) the crossroad.

To define the specification, it is natural that a train is allowed to enter the crossroad only if its traffic light is green. The purpose of the entry and exit points \( x_i, i = 1, 2, 3, 4 \), is

![Fig. 4 A railway crossroad](image-url)
to allow a limited number of trains in the crossroad area from the direction of the green light. The light can turn red at any moment, but the other traffic light can be set to green only if all the trains have left the crossroad area. In this example, we consider the case where at most three trains are allowed to enter the crossroad area on one green light. For this purpose, the entry points must also be controllable to protect another train to enter. This part of the specification is modeled by buffers depicted in Fig. 6. Another part of the specification governs the behavior of the traffic lights. First, both lights must be red before one of the traffic lights is set to green, stay green for a while, and then must be set to red again. The traffic lights should take turns, so that no trains are waiting for ever, see Fig. 7. For simplicity, we do not model the mechanism (such as a clock) that sets the traffic lights to green for a specific amount of time units. The overall specification is then depicted in Fig. 8. The set of uncontrollable events is thus $\Sigma_u = \{x_2, x_4\}$; all other events are controllable.

To make the specification controllable with respect to $\Sigma_1, \Sigma_2, \text{ and } \Sigma_k$ (where $\Sigma_k$ is initialized to the empty set), we need to take $\Sigma_k = \{g_1, g_2, r_1\}$. Now we can compute the coordinator as the projection $P_k(G_1) \parallel P_k(G_2)$, and the languages sup $C_k$, sup $C_{1+k}$ and sup $C_{2+k}$ as defined in (*), see Figs. 9, 10, and 11. It can be verified that sup $C_k \subseteq P_k(\text{sup } C_{i+k})$, for $i = 1, 2$, hence Theorem 6 applies and the result (that is, in the monolithic notation, the language sup $C_{1+k} \parallel \text{sup } C_{2+k}$) is the supremal conditionally-controllable sublanguage of the specification, cf. Fig. 12. Note that the difference with the specification is the correct marking of the states.

6 Coordinator for nonblockingness

So far, we have only considered a coordinator for safety. In this section, we discuss a coordinator for nonblockingness. To this end, we first prove a fundamental theoretical result and then give an algorithm to construct a coordinator for nonblockingness.

Recall that a generator $G$ is nonblocking if $L_m(G) = L(G)$.

**Theorem 6** Consider languages $L_1$ over $\Sigma_1$ and $L_2$ over $\Sigma_2$, and let the projection $P_0 : (\Sigma_1 \cup \Sigma_2)^* \rightarrow \Sigma_0^*$, with $\Sigma_1 \cap \Sigma_2 \subseteq \Sigma_0$, be an $L_i$-observer, for $i = 1, 2$. Let $G_0$ be

![Diagram](image-url)
a nonblocking generator with $L_m(G_0) = P_0(L_1) \parallel P_0(L_2)$. Then the composed language $L_1 \parallel L_2 \parallel L_m(G_0)$ is nonblocking, that is, $L_1 \parallel L_2 \parallel L_m(G_0) = L_1 \parallel L_2 \parallel L_m(G_0)$.

Proof Let $L_0 = L_m(G_0)$. By Lemma 1, $L_1 \parallel L_2 \parallel L_0 = L_1 \parallel L_2 \parallel \overline{L_0}$ if and only if $P_0(L_1) \parallel P_0(L_2) \parallel L_0 = P_0(L_1) \parallel P_0(L_2) \parallel L_0$.

However, for our choice of the coordinator, this equality always holds because both sides of the later equation are $\overline{L_0}$.

This result is demonstrated in the following example.

Example 6 Consider two nonblocking generators $G_1$ and $G_2$ depicted in Fig. 13. Their synchronous product is shown in Fig. 14. One can see that the generator $G_1 \parallel G_2$ is blocking because no marked state is reachable from state 3. It can be verified that the projection $P : \{a, b, c, d\}^* \rightarrow \{a, b, d\}^*$ is an $L(G_1)$- and $L(G_2)$-observer. The generator $G_0$ is then a nonblocking (trimmed) part of the synchronous product $P(G_1) \parallel P(G_2)$ of generators depicted in Fig. 15, that is $L_m(G_0) = \{a\}$, and the synchronous product of $G_1 \parallel G_2$ with $G_0$ is shown in Fig. 16. One can see that the result is nonblocking. It is important to notice that event $b$ belongs to the event set of the generator $G_0$.
The previous example shows that even though the result is nonblocking, it is disputable whether such a coordinator is acceptable. If we assume that event $b$ is uncontrollable, then the coordinator prevents an uncontrollable event from happening and the result depicted in Fig. 16 is not controllable with respect to the plant depicted in Fig. 14. Although it is not explicitly stated that a coordinator is not allowed to do so, we further discuss this issue and suggest a solution useful in our coordination control framework.

In general, local supervisors $\sup C_1 + k$ and $\sup C_2 + k$ computed in Section 5 might be blocking. However, we can always choose the language

$$L_C = \sup C(\sup C_1 + k) \parallel \sup C_2 + k), \quad P_0(\sup C_1 + k) \parallel P_0(\sup C_2 + k), \quad \Sigma_{0, u}),$$

where the projection $P_0$ is a $\sup C_{i+k}$-observer, for $i = 1, 2$. The following result shows that the language $\sup C_1 + k \parallel \sup C_2 + k \parallel L_C$ is nonblocking and controllable.

**Theorem 7** Consider the notation as defined in Problem 3, Algorithm 2, (⋆), and (2). Then the language

$$\sup C_1 + k \parallel \sup C_2 + k \parallel L_C = \sup C_1 + k \parallel \sup C_2 + k \parallel L_C$$

is controllable with respect to the plant language $L(G_1) \parallel L(G_2)$.

**Proof** To prove nonblockingness, we use Lemma 11 in two steps. Namely, $\sup C_{i+k} \parallel L_C$ if and only if $P_0(\sup C_{i+k}) \parallel L_C = P_0(\sup C_{i+k}) \parallel L_C$, for $i = 1, 2$, which always holds because both sides of the later equation are equal to $L_C$. Using Lemma 11 again,

$$\sup C_{i+k} \parallel L_C \parallel \sup C_{j+k} \parallel L_C = \sup C_{i+k} \parallel L_C \parallel \sup C_{j+k} \parallel L_C$$

if and only if

$$P_0(\sup C_{i+k} \parallel L_C) \parallel P_0(\sup C_{j+k} \parallel L_C) = P_0(\sup C_{i+k} \parallel L_C) \parallel P_0(\sup C_{j+k} \parallel L_C)$$

(3) because if the projection $P_0$ is a $\sup C_{i+k}$-observer, for $i = 1, 2$, and an $L_C$-observer (since it is an identity), then the projection $P_0$ is also a $\sup C_{i+k} \parallel L_C$-observer by Pena et al. (2009). But (3) always holds because $P_0(\sup C_{i+k} \parallel L_C) = P_0(\sup C_{i+k}) \parallel L_C = L_C$, by
Fig. 11  Supervisor sup $C_{2+k}$

Fig. 12  The supremal conditionally-controllable sublanguage $\text{sup } C_{1+k} \parallel \text{sup } C_{2+k}$

Fig. 13  Generators $G_1$ and $G_2$

Fig. 14  Synchronous product $G_1 \parallel G_2$

Fig. 15  Generators $P(G_1)$ and $P(G_2)$
Lemma 9, hence both sides are equal to $\overline{L_C}$. Thus, summarized, we have that

$$\sup C_{1+k} \parallel \sup C_{2+k} \parallel L_C = \sup C_{1+k} \parallel L_C \parallel \sup C_{2+k} \parallel L_C$$

$$= \sup C_{1+k} \parallel L_C \parallel \sup C_{2+k} \parallel L_C$$

$$= \sup C_{1+k} \parallel \sup C_{2+k} \parallel L_C.$$

To prove controllability, note that $\sup C_{i+k}$ is controllable with respect to $\overline{L_C}$, for $i = 1, 2$, and $L_C$ is controllable with respect to $P_0(\sup C_{1+k}) \parallel P_0(\sup C_{2+k})$. By Lemma 7 used several times, and the nonconflictness shown above, we obtain that

$$- \sup C_{1+k} \parallel L_C \text{ is controllable with respect to } (\sup C_{i+k}) \parallel (P_0(\sup C_{1+k}) \parallel P_0(\sup C_{2+k})), \text{ for } i = 1, 2,$$

$$- (\sup C_{1+k} \parallel L_C) \parallel (\sup C_{2+k} \parallel L_C) = \sup C_{1+k} \parallel \sup C_{2+k} \parallel L_C \text{ is controllable with respect to }$$

$$(\sup C_{1+k} \parallel P_0(\sup C_{1+k}) \parallel P_0(\sup C_{2+k})) \parallel (\sup C_{2+k} \parallel P_0(\sup C_{1+k}) \parallel P_0(\sup C_{2+k}))$$

that can be simplified to $\sup C_{1+k} \parallel \sup C_{2+k}$,

$$- \sup C_{1+k} \parallel \sup C_{2+k} \text{ is controllable with respect to } (L(G_1) \parallel \sup C_k) \parallel$$

$$(L(G_1) \parallel \sup C_k) = L(G_1) \parallel L(G_2) \parallel \sup C_k \parallel \sup C_k,$$

$$- L(G_1) \parallel L(G_2) \parallel \sup C_k \text{ is controllable with respect to } L(G_1) \parallel L(G_2) \parallel L(G_k) \text{ because the language sup C_k is controllable with respect to L(G_k).}$$

Using transitivity of controllability, Lemma 8, we obtain that $\sup C_{1+k} \parallel \sup C_{2+k} \parallel L_C$ is controllable with respect to $L(G_1) \parallel L(G_2) \parallel L(G_k) = L(G_1) \parallel L(G_2)$, because the coordinator $G_k$ is constructed in such a way that it does not change the plant.

We demonstrate this improvement in the following example.

**Example 7** Consider the generators of Example 6. Note that $G_1 \parallel G_2 \parallel G_0$, Fig. 16, is not controllable with respect to the plant $G_1 \parallel G_2$, Fig. 14, if $b$ is uncontrollable. The generator $G_0 = P(G_1) \parallel P(G_2)$ is depicted in Fig. 17. It is not hard to see that if $b$ is not controllable, then the supremal controllable sublanguage of $L_m(G_0)$ with respect to $L(G_0)$ is $L_C = \{\epsilon\}$, because event $a$ must be prevented from happening. Therefore, the language of $L(G_1 \parallel G_2) \parallel L_C = \{\epsilon\}$ as expected.

![Fig. 16 Synchronous product $G_1 \parallel G_2 \parallel G_0$](image1)

![Fig. 17 Generator $P(G_1) \parallel P(G_2)$](image2)
We can now summarize this method as an algorithm.

**Algorithm 9 (Coordinator for nonblockingness)** Consider the notation above.

1. Compute \( \sup C_1 + k \) and \( \sup C_2 + k \) as defined in (*)
2. Let \( \Sigma_0 := \Sigma_k \) and \( P_0 := P_k \).
3. Extend the event set \( \Sigma_0 \) so that the projection \( P_0 \) is both a \( \sup C_1 + k \) and a \( \sup C_2 + k \)-observer.
4. Define the coordinator \( C \) as the minimal nonblocking generator such that \( L_m(C) = \sup C(P_0(\sup C_1 + k) \parallel P_0(\sup C_2 + k), P_0(\sup C_1 + k) \parallel P_0(\sup C_2 + k), \Sigma_0, u) \).

This algorithm (Step 1) is based on the computation of the languages \( \sup C_1 + k \) and \( \sup C_2 + k \) defined in (*), which can be computed using a standard algorithm for the computation of supremal controllable sublanguages. If the assumption of Theorem 6 is satisfied, the computed languages are the languages of local supervisors that are the candidates to solve the problem. However, the composition \( \sup C_1 + k \parallel \sup C_2 + k \) can be blocking, and a coordinator for nonblockingness is required.

In Step 2, we define a new event set \( \Sigma_0 \) (and the corresponding projection) that is initialized to be the event set \( \Sigma_k \) used in the computation in Step 1.

In Step 3 of the algorithm, the event set \( \Sigma_0 \) must be extended so that the projection \( P_0 \) is both a \( \sup C_1 + k \) and \( \sup C_2 + k \)-observer. Thus, in consequence of the extension operation, \( \Sigma_k \) can become a proper subset of \( \Sigma_0 \). Even though the computation of such a minimal extension is NP-hard, a polynomial algorithm computing a reasonable extension exists, cf. Feng and Wonham (2010) for more details and the algorithm.

Finally, in Step 4, the coordinator generator \( C \) is defined as a minimal nonblocking generator accepting the supremal controllable sublanguage of the language \( P_0(\sup C_1 + k) \parallel P_0(\sup C_2 + k) \) with respect to the language \( P_0(\sup C_1 + k) \parallel P_0(\sup C_2 + k) \). This idea has been used by Feng in (2007). In other words, if \( S_1 \) and \( S_2 \) are generators for languages \( \sup C_1 + k \) and \( \sup C_2 + k \), respectively, then the coordinator \( C \) is computed as the generator for the supremal controllable sublanguage of \( P_0(S_1) \parallel P_0(S_2) \). Since \( P_0 \) is an observer, the computation can be done in polynomial time, cf. Wonham (2012).

**Remark 1** In the previous section we discussed the case when \( \sup C_k \not\subseteq P_k(\sup C_i + k) \), for \( i = 1, 2 \). Note that the coordinator \( L_C \) discussed in this section can also be used in that case because \( \sup C_1 + k \parallel L_C \) and \( \sup C_2 + k \parallel L_C \) then form synchronously nonconflicting local supervisors such that their overall behavior is controllable with respect to the global plant. Hence, although this solution may not be optimal, it presents a solution in the case of (non-prefix-closed) languages that do not satisfy the assumptions of Theorem 6, or of those of Section 7 in the case of prefix-closed languages.

7 Supremal prefix-closed languages

In this section, we revise the case of prefix-closed languages. We use the local control consistency property (LCC) instead of the output control consistency property (OCC), cf. (Komenda et al. 2012c). The reason for this is that LCC is a less restrictive condition than OCC, as shown in Schmidt and Breindl (2011, Lemma 4.4). Moreover, the extension of our approach to an arbitrary number of local plants is sketched.
Theorem 8 Let \( K \) be a prefix-closed sublanguage of the plant language \( L = L(G_1 \| G_2 \| G_k) \) where \( G_i \) is a generator over \( \Sigma_i \), for \( i = 1, 2, k \). Assume that the language \( K \) is conditionally decomposable, and define the languages \( \sup C_k, \sup C_{1+k}, \sup C_{2+k} \) as in (*)]. Let the projection \( P_{i+k} \) be an \((P_{i+k})^{-1}(L(G_i))\)-observer and LCC for the language \((P_{i+k})^{-1}(L(G_i)), for \( i = 1, 2 \). Then

\[
\sup C_{1+k} \parallel \sup C_{2+k} = \sup \text{cC}(K, L, (\Sigma_{1,u}, \Sigma_{2,u}, \Sigma_{k,u})).
\]

Proof In this proof, let \( \sup \text{cC} \) denote the supremal conditionally controllable language \( \sup \text{cC}(K, L, (\Sigma_{1,u}, \Sigma_{2,u}, \Sigma_{k,u})) \), and \( M \) the parallel composition \( \sup C_{1+k} \parallel \sup C_{2+k} \). It is shown in Komenda et al. (2012c, Theorem 11) that \( \sup \text{cC} \) is a subset of \( M \) and that \( M \) is a subset of \( K \). To prove that \( P_k(M) \Pi_{k,u} \cap L(G_k) \) is a subset of \( P_k(M) \), consider a word \( x \in P_k(M) \) and an uncontrollable event \( a \in \Sigma_{k,u} \) such that the word \( xa \in L(G_k) \). To show that the word \( xa \) is in \( P_k(M) \), \( P_k(M) = P_{k+1+k}(\sup C_{1+k}) \cap P_{k+2+k}(\sup C_{2+k}) \), note that there exists a word \( w \in M \) such that \( P_k(w) = x \). It is shown in Komenda et al. (2012c, Theorem 11) that there exists a word \( u \in (\Sigma_1 \setminus \Sigma_k)^* \) such that the word \( P_{1+k}(w)ua \in (P_{1+k})^{-1}(L(G_1)) \) and the word \( P_{1+k}(w) \) is in \( L(G_1) ) \parallel \sup C_k \). As the projection \( P_{k+1+k} \) is LCC for the language \((P_{1+k})^{-1}(L(G_1)) \), there exists a word \( u' \in (\Sigma_u \setminus \Sigma_k)^* \) such that \( P_{1+k}(w)u' \) is in \((P_{1+k})^{-1}(L(G_1)) \). Then, controllability of \( \sup C_{1+k} \) implies that \( P_{1+k}(w)u' \) is in \( \sup C_{1+k} \), that is, \( xa \) is in \( P_{k+1+k}(\sup C_{1+k}) \). Analogously, we can prove that \( xa \) is in \( P_{k+2+k}(\sup C_{2+k}) \). Thus, \( xa \) is in \( P_k(M) \). The rest of the proof is the same as in (Komenda et al. 2012c, Theorem 11).

In this Theorem, a relatively large number of properties that the coordinator, the local plants and the specification have to satisfy is assumed. However, a polynomial algorithm extending the coordinator event set so that the language \( K \) becomes conditionally decomposable has already been discussed, see Komenda et al. (2012b). In addition, to ensure that the projection \( P_{i+k} \) is an \((P_{i+k})^{-1}(L(G_i))\)-observer and LCC for the language \((P_{i+k})^{-1}(L(G_i)), for \( i = 1, 2 \), the coordinator event set can again be extended so that the conditions are fulfilled (Schmidt and Breindl 2011; Feng and Wonham 2010).

Conditions of Theorem 10 imply that the projection \( P_k \) is LCC for the language \( L \).

Lemma 6 Let \( G_i \) over \( \Sigma_i \) be generators, for \( i = 1, 2 \). Let \( \Sigma = \Sigma_1 \cup \Sigma_2 \), and let \( P_1 : \Sigma^* \to \Sigma_i^* \), for \( i = 1, 2, k \) and \( \Sigma_k \subseteq \Sigma \), be projections. If \( \Sigma_1 \cap \Sigma_2 \) is a subset of \( \Sigma_k \) and the projection \( P_{i+k} \) is LCC for the language \((P_{i+k})^{-1}(L(G_i)), for \( i = 1, 2 \), then the projection \( P_k \) is LCC for the language \( L = L(G_1 \| G_2 \| G_k) \).

Proof For a word \( s \) in \( L \) and an event \( \sigma_u \) in \( \Sigma_{k,u} \), assume that there exists a word \( u \) in \( (\Sigma \setminus \Sigma_k)^* \) such that \( su \sigma_u \in L \). Then \( P_{i+k}(su \sigma_u) = P_{i+k}(s) P_{i+k}(u) \), \( \sigma_u \) in \((P_{i+k})^{-1}(L(G_i)) \) implies that there exists a word \( v_i \in (\Sigma_{i+k,u} \setminus \Sigma_k)^* \), for \( i = 1, 2 \), such that \( P_{i+k}(s)v_i \sigma_u \) in \((P_{i+k})^{-1}(L(G_i)) \). As \( P_k(v_i) = \varepsilon \), \( P_k(v_i) = v_i \) and we get that \( P_k(v_i) P_k(\sigma_u) \) in \( L(G_i) \), for \( i = 1, 2, k \). Consider a word \( u' \) in \( (\{v_1\} \cup \{v_2\}) \). Then \( P_k(u') = v_i \) and, thus, \( su' \sigma_u \) in \( L \). Moreover, \( u' \) is in \((\Sigma_u \setminus \Sigma_k)^* \).
our coordination control architecture coincides with the global optimal solution given by
the supremal controllable sublanguage of the specification.

Theorem 9 Consider the setting of Theorem 10. If, in addition, \(L(G_k)\) is a subset of \(P_k(L)\)
and the projection \(P_{i+k}\) is LCC for the language \(L\), for \(i = 1, 2\), then

\[
\sup C(K, L, \Sigma_{iu}) = \sup cC(K, L, (\Sigma_{1,u}, \Sigma_{2,u}, \Sigma_{k,u})).
\]

Proof It was shown in Komenda et al. (2012c, Theorem 15) that the projection \(P_k\) is an
\(L\)-observer. Moreover, by Lemma 6, the projection \(P_k\) is LCC for the language
\(L\). Denote \(\sup C = \sup C(K, L, \Sigma_{iu})\). We prove that the language \(P_k(\sup C)\) is controllable with
respect to \(L(G_k)\). Consider a word \(t\) in \(P_k(\sup C)\) and an event \(a\) in \(\Sigma_{k,u}\) such that the
word \(ta\) is in \(L(G_k)\), which is a subset of \(P_k(L)\). We proved in (Komenda et al. 2012c,
Theorem 15) that there exist words \(s\) in \(\sup C\) and \(u\) in \((\Sigma \setminus \Sigma_k)^*\) such that \(sua\) is in \(L\)
and \(P_k(sua) = ta\). By the LCC property of the projection \(P_k\), there exists a word \(u'\) in
\((\Sigma_u \setminus \Sigma_k)^*\) such that \(su'a\) is in \(L\). By controllability of the language \(\sup C\) with respect
to \(L\), the word \(su'a\) is in \(\sup C\), that is, \(P_k(su'a) = ta\) is in \(P_k(\sup C)\). Thus, (1) of
Definition 3 holds. By Komenda et al. (2012c, Theorem 15), the projection \(P_{i+k}\) is an \(L\)-
observer, for \(i = 1, 2\). To prove (2) of Definition 3, consider a word \(t\) in \(P_{i+k}(\sup C)\), for
\(1 \leq i \leq 2\), and an event \(a\) in \(\Sigma_{i+k,u}\) such that the word \(ta\) is in \(L(G_i) \parallel P_k(\sup C)\). We
proved in Komenda et al. (2012c, Theorem 15) that there exist words \(s\) in \(\sup C\) and \(u\) in
\((\Sigma \setminus \Sigma_k)^*\) such that \(sua\) is in \(L\) and \(P_{i+k}(sua) = ta\). As the projection \(P_{i+k}\) is LCC for
the language \(L\), there exists a word \(u'\) in \((\Sigma_u \setminus \Sigma_{i+k})^*\) such that \(su'a\) is in \(L\). Then con-
trollability of \(\sup C\) with respect to \(L\) implies that \(su'a\) is in \(\sup C\), that is, \(P_{i+k}(su'a) = ta\) is in \(P_{i+k}(\sup C)\). The other inclusion is the same as in Komenda et al. (2012c,
Theorem 15).

Finally, a natural and simple extension to more than two local subsystems with one cen-
tral coordinator is sketched. All concepts and results carry over to this general case of \(n\)
subsystems, where the coordinator event set \(\Sigma_k\) should contain all shared events (events
common to two or more subsystems). Conditional decomposability is then simply decom-
posability with respect to event sets \((\Sigma_i)_{i=1}^n\) and \(\Sigma_k\), cf. Section 3. It is a very good news
for large systems that conditional decomposability can be checked in polynomial time with
respect to the number of components as has been noticed in Komenda et al. (2012b). Note
that unlike the previous form of conditional controllability, Definition 3 can be extended
to the general case of \(n\) subsystems in an obvious way. Namely, conditions (2) and (3) are
replaced by \(n\) conditions of the form \(P_{i+k}(K)\) is controllable with respect to \(L(G_i) \parallel P_k(K)\)
and \(\Sigma_{i+k,u}\).

8 Conclusion

We have revised, simplified, and extended the coordination control scheme for discrete-
event systems. These results have been used, for the case of prefix-closed languages, in
the implementation of the coordination control plug-in for libFAUDES. We have identified
cases, where supremal conditionally-controllable sublanguages can be computed even in
the case of non-prefix-closed specification languages, and proposed coordinators for non-
blockingness in addition to coordinators for safety developed in our earlier publications.
Note that a general procedure for the computation of supremal conditionally-controllable sublanguages in the case of non-prefix-closed specification languages is still missing.

In our original approach to coordination control (Komenda et al. 2011a, b, 2012c), we have established a list of conditions required to compute the local supervisors and the supervisor for the coordinator so that the composition of these supervisors achieves the globally optimal solution (the supremal controllable sublanguage). These conditions are restrictive although our approach is general in the sense that the conditions can be met by enlarging the coordinator event set. There exist examples where too many events have to be added to the coordinator event set to make the conditional decomposability, observer, and LCC conditions hold. That has been the main motivation to find alternative conditions that would enable the synthesis of globally optimal supervisors using coordinated local supervisors. This condition is stated in Theorem 6. Note that the required inclusion means in fact that the equality holds as the other inclusion is always true as shown in Lemma 2—the local supervisor that has the coordinator supervisor in its plant language does always at least as much “disabling” job as the supervisor for the coordinator itself. Intuitively, the sufficient condition of Theorem 6 implies that the “disabling” job done by the supervisor for the coordinator cannot be improved by the local supervisors \( \sup C_{i+k}, \quad i = 1, 2, \) which means that these supervisors disable only events from the sets \( \Sigma_1 \setminus \Sigma_k \) and \( \Sigma_2 \setminus \Sigma_k \) that are always disjoint due to conditional independence property. It is well known that modular supervisory control works well if local modules (components) are over disjoint event sets—both nonblocking-ness and supremality can then be achieved locally. It is not surprising that the inclusion of Theorem 6 ensures that the supremal conditionally-controllable sublanguage can be computed without any further condition even in the non-prefix-closed case. In addition, the condition of Theorem 6 is often satisfied in practical examples, as documented by the paint factory example (Boutin and van Schuppen 2011) or by Example 5. If this condition holds, then the observer and LCC conditions are not needed anymore. We emphasize here that the computation of the globally supremal solution is sometimes difficult (especially in the non-prefix-closed case) and a non-optimal solution will be a reasonable compromise in many cases, especially if none of the sufficient conditions can be achieved without having to extend the coordinator event set too much. Such a non-optimal (but controllable and non-blocking solution) can always be computed using the approach of Theorem 8 described in Algorithm 9.

Another aspect that requires further investigation is the generalization of coordination control from the current case of one central coordinator to multilevel coordination control with several coordinators on different levels. In fact, one central coordinator is typically not enough in case of a large number of local subsystems, because too many events must be communicated (added into the coordinator event set) between the coordinator and local subsystems. This general architecture will be computationally more efficient, because less events need to be communicated. In the multi-level coordination control the subsystems will be organized into different groups and each group will have a coordinator meaning that only events from a given group will be communicated among subsystems of the same group via the coordinator. An interested reader is referred to Komenda et al. (2013) for more details.

Acknowledgments The research of two first authors was supported by RVO: 67985840. In addition, the first author was supported by the Grant Agency of the Czech Republic under grant P103/11/0517 and the second author by the Grant Agency of the Czech Republic under grant P202/11/P028. The authors gratefully acknowledge very useful suggestions and comments of the anonymous referees.

Acknowledgments The research of two first authors was supported by RVO: 67985840. In addition, the first author was supported by the Grant Agency of the Czech Republic under grant P103/11/0517 and the second author by the Grant Agency of the Czech Republic under grant P202/11/P028. The authors gratefully acknowledge very useful suggestions and comments of the anonymous referees.
Appendix Auxiliary results

In this section, we list auxiliary results required in the paper.

Lemma 7 (Proposition 4.6, (Feng 2007)) Let $L_i$ over $\Sigma_i$, for $i = 1, 2$, be prefix-closed languages, and let $K_i$ be a controllable sublanguage of $L_i$ with respect to $L_i$ and $\Sigma_{i,u}$. Let $\Sigma = \Sigma_1 \cup \Sigma_2$. 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 $\Sigma_u$.

Lemma 8 (Komenda et al. 2012c) Let $K$ be a subset of a language $L$, and $L$ be a subset of a language $M$ over $\Sigma$ such that $K$ is controllable with respect to $\overline{L}$ and $\Sigma_u$, and $L$ is controllable with respect to $\overline{M}$ and $\Sigma_u$. Then $K$ is controllable with respect to $\overline{M}$ and $\Sigma_u$.

Lemma 9 (Wonhan 2012) Let $P_k : \Sigma^* \rightarrow \Sigma_k^*$ be a projection, and let $L_i$ be a language over $\Sigma_i$, where $\Sigma_i$ is a subset of $\Sigma$, for $i = 1, 2$, and $\Sigma_1 \cap \Sigma_2$ is a subset of $\Sigma_k$. Then $P_k(L_1 \parallel L_2) = P_k(L_1) \parallel P_k(L_2)$.

Lemma 10 (Komenda et al. 2012c) Let $L_i$ be a language over $\Sigma_i$, for $i = 1, 2$, and let $P_i : (\Sigma_1 \cup \Sigma_2)^* \rightarrow \Sigma_i^*$ be a projection. Let $A$ be a language over $\Sigma_1 \cup \Sigma_2$ such that $P_1(A)$ is a subset of $L_1$ and $P_2(A)$ is a subset of $L_2$. Then $A$ is a subset of $L_1 \parallel L_2$.

Lemma 11 (Pena et al. 2009) Let $L_i$ be a language over $\Sigma_i$, for $i \in J$, and let $\bigcup_{k \neq \ell} (\Sigma_k \cap \Sigma_\ell) \subseteq \Sigma_0$. If $P_{1,0} : \Sigma_{i_0}^* \rightarrow (\Sigma_i \cap \Sigma_0)^*$ is an $\ell_i$-observer, for $i \in J$, then $\| \bigcap_{i \in J} L_i = \| \bigvee_{i \in J} P_{1,0}(L_i)$ if and only if $\| \bigcap_{i \in J} P_{1,0}(L_i) = \| \bigvee_{i \in J} P_{1,0}(L_i)$.

Lemma 12 (Komenda et al. 2011b) A language $K \subseteq (\Sigma_1 \cup \Sigma_2 \cup \ldots \cup \Sigma_n)^*$ is conditionally decomposable with respect to event sets $\Sigma_1$, $\Sigma_2$, $\ldots$, $\Sigma_n$. $\Sigma_k$ if and only if there exist languages $M_{i+k} \subseteq \Sigma_{i+k}^*$, $i = 1, 2, \ldots, n$, such that $K = \bigcup_{i=1}^n M_{i+k}$.

References

Barrett G, Lafortune S (2000) Decentralized supervisory control with communicating controllers. IEEE Trans Autom Control 45(9):1620–1638
Boutin ODM, van Schuppen JH (2011) On the control of the paint factory scale model. CWI Technical Report MAC-1103. Available at http://oai.cwi.nl/oai/asset/18598/18598D.pdf
Bravo HJ, Da Cunha AEC, Pena P, Malik R, Cury JER (2012) Generalised verification of the observer property in discrete event systems. In: Proceedings of WODES 2012. Guadalajara, Mexico, pp 337–342
Cassandras CG, Lafortune S (2008) Introduction to discrete event systems, 2nd edn, Springer
Feng L (2007) Computationally efficient supervisor design for discrete-event systems. Ph.D. thesis, University of Toronto
Feng L, Wonham W (2010) On the computation of natural observers in discrete-event systems. Discret Event Dyn Syst 20:63–102
Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of np-completeness. In: Freeman, W H
Gaudin B, Marchand H (2004) Supervisory control of product and hierarchical discrete event systems. Eur J Control 10(2):131–145
Komenda J, Masopust T, van Schuppen JH (2011a) Coordinated control of discrete event systems with non-prefix-closed languages. In: Proceedings of IFAC world congress 2011, Milan, Italy, pp 6982–6987
Komenda J, Masopust T, van Schuppen JH (2011b) Synthesis of controllable and normal sublanguages for discrete-event systems using a coordinator. Syst Control Lett 60(7):492–502
Komenda J, Masopust T, van Schuppen JH (2012a) On algorithms and extensions of coordination control of discrete-event systems. In: Proceedings of WODES 2012, Guadalajara, Mexico, pp 245–250
Komenda J, Masopust T, van Schuppen JH (2012b) On conditional decomposability. Syst Control Lett 61(12):1260–1268
Komenda J, Masopust T, van Schuppen JH (2012c) Supervisory control synthesis of discrete-event systems using a coordination scheme. Automatica 48(2):247–254
Komenda J, Masopust T, van Schuppen JH (2013) Multilevel coordination control of modular DES. In: Proceedings of 52nd IEEE conference on decision and control (CDC)
Komenda J, van Schuppen JH (2008) Coordination control of discrete event systems. In: Proceedings of WODES 2008, Gothenburg, Sweden, pp 9–15
Komenda J, van Schuppen JH, Gaudin B, Marchand H (2008) Supervisory control of modular systems with global specification languages. Automatica 44(4):1127–1134
Kumar R, Takai S (2007) Inference-based ambiguity management in decentralized decision-making: decentralized control of discrete event systems. IEEE Trans Autom Control 52(10):1783–1794
Leduc RJ, Lawford M, Wonham WM (2005) Hierarchical interface-based supervisory control-part ii: parallel case. IEEE Trans Autom Control 50(9):1336–1348
Moor T et al (2012) libFAUDES – a discrete event systems library. Available at http://www.rt.eei.unierlangen.de/FGdes/faudes/
Pena P, Cury J, Lafortune S (2008) Polynomial-time verication of the observer property in abstractions. In: Proceedings of ACC 2008, Seattle, pp 465–470
Pena PN, Cury JER, Lafortune S (2009) Verification of nonconflict of supervisors using abstractions. IEEE Trans Autom Control 54(12):2803–2815
Ramadge PJ, Wonham WM (1987) Supervisory control of a class of discrete event processes. SIAM J Control Optim 25(1):206–230
Ricker SL, Rudie K (2000) Know means no: incorporating knowledge into discrete-event control systems. IEEE Trans Autom Control 45(9):1656–1668
Rudie K, Wonham WM (1992) Think globally, act locally: decentralized supervisory control. IEEE Trans Autom Control 37(11):1692–1708
Schmidt K, Breindl C (2008) On maximal permissiveness of hierarchical and modular supervisory control approaches for discrete event systems. In: Proceedings of WODES 2008, Gothenburg, pp 462–467
Schmidt K, Breindl C (2011) Maximally permissive hierarchical control of decentralized discrete event systems. IEEE Trans Autom Control 56(4):723–737
Schmidt K, Moor T, Perk S (2008) Nonblocking hierarchical control of decentralized discrete event systems. IEEE Trans Autom Control 53(10):2252–2265
Su R, van Schuppen JH, Rooda JE (2010) Model abstraction of nondeterministic finite-state automata in supervisor synthesis. IEEE Trans Autom Control 55(11):2527–2541
Su R, van Schuppen JH, Rooda JE (2012) Maximally permissive coordinated distributed supervisory control of nondeterministic discrete-event systems. Automatica 48(7):1237–1247
Wong K (1998) On the complexity of projections of discrete-event systems. In: Proceedings of WODES 1998, Cagliari, pp 201–206
Wong K, Wonham W (1996) Hierarchical control of discrete-event systems. Discret Event Dyn Syst 6(3):241–273
Wonham WM (2012) Supervisory control of discrete-event systems. Lecture notes, University of Toronto. Available at http://www.control.utoronto.ca/DES/
Yoo T, Lafortune S (2002) A general architecture for decentralized supervisory control of discrete-event systems. Discret Event Dyn Syst 12(3):335–377
Yoo T, Lafortune S (2004) Decentralized supervisory control with conditional decisions: supervisor existence. IEEE Trans Autom Control 49(11):1886–1904
Zhong H, Wonham WM (1990) On the consistency of hierarchical supervision in discrete-event systems. IEEE Trans Autom Control 35(10):1125–1134
Jan Komenda received M.Sc. in mathematics from the Masaryk University (Brno), Czech Republic, in 1994, and was awarded a Ph.D. degree in control and computer science from Université de Franche-Comté (Besançon), France in 1999. He worked at CWI Amsterdam (The Netherlands) in 2001-2003 in the group of J.H. van Schuppen. He is a researcher at the Institute of Mathematics of the Academy of Sciences of the Czech Republic, Brno, Czech Republic. His current research interests are in supervisory control of logical and timed discrete event systems using methods of coalgebra and idempotent algebra. He is Associated Editor of Automatica.

Tomáš Masopust received M.Sc. in computer science from the Masaryk University, Brno, Czech Republic in 2004, and Ph.D. in computer science from the Brno University of Technology, Brno, Czech Republic in 2007. In 2010-2011 he worked at CWI Amsterdam, The Netherlands, in the group of J.H. van Schuppen, and in 2012-2013 at the University of Bayreuth, Germany, in the Theoretical computer science group. He is a researcher at the Institute of Mathematics of the Academy of Sciences of the Czech Republic, Brno, Czech Republic. His research interest includes supervisory control of discrete-event systems and theoretical computer science.
Jan H. van Schuppen is the director and researcher at the company Van Schuppen Control Research in Amsterdam, The Netherlands. He is Professor Emeritus at the Department of Mathematics, Delft University of Technology, Delft, The Netherlands. He studied at the Department of Applied Physics of Delft University of Technology and was awarded a Ph.D. diploma by the Department of Electrical Engineering and Computer Science of the University of California at Berkeley, CA, U.S.A. in 1973 where his research advisor was Pravin Varaiya. Van Schuppen’s research interests include control of discrete-event systems and of hybrid systems, stochastic control, realization, and system identification. In discrete-event systems his interests are primarily: system theory, the use of coalgebra, decentralized control, and control and diagnosis of asynchronous systems. In applied research his interests include engineering problems of control of motorway traffic, of communication networks, and modeling, identification, and control in the life sciences. He is Co-Editor of the journal Mathematics of Control, Signals, and Systems, and was Department Editor of the journal Discrete Event Dynamic Systems.