Coordinated Control of Discrete Event Systems with Nonprefix-Closed Languages

Jan Komenda [*], **Tomáš Masopust** [**], **Jan H. van Schuppen** [**]

[*] *Institute of Mathematics, Czech Academy of Sciences*
Žižkova 22, 616 62 Brno, Czech Republic
(e-mail: komenda@ipm.cz)
[**] *CWI, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands*
(e-mails: T.Masopust@cwi.nl, J.H.van.Schuppen@cwi.nl)

Abstract: In this paper, the supervisory control synthesis of modular or distributed discrete-event systems is discussed. The coordination-control architecture proposed by Komenda and van Schuppen (2008) and studied for prefix-closed specification languages by Komenda et al. (2010a,b) is generalized to the case of non-prefix-closed global specification languages and non-prefix-closed plant languages. Necessary and sufficient conditions for the solvability of the coordinated supervisory control synthesis problem are characterized in terms analogous to the terms of controllable and closed languages used in supervisory control theory of monolithic discrete-event systems.

Keywords: Discrete-event systems, supervisory control, distributed control, controllability.

1. INTRODUCTION

A modular or a distributed discrete-event system is a synchronous product of two or more modules or subsystems. In supervisory control of a distributed discrete-event system each module or subsystem has its own observation channel and a supervisor. The local (also called modular or decentralized) control synthesis then consists of synthesizing local supervisors for each subsystem. The global supervisor formally consists of the synchronous product of local supervisors although this product is not computed in practice.

Given a rational global specification language (i.e., a language recognizable by a finite-state machine), the supremal controllable sublanguage from which the optimal supervisor is built can always be computed. Such a global control synthesis of a distributed discrete-event system consists in computing the global plant (a synchronous product of local plants), and the control synthesis is then carried out in the usual way. However, the global control synthesis is not always possible because of state complexity reasons. On the other hand, the purely local control synthesis does not work in general. Therefore, Komenda and van Schuppen (2008) have proposed a coordinated control architecture as a trade-off between the purely local control synthesis and the global control synthesis.

In this paper, the problem of the supervisory control synthesis of modular or distributed discrete-event systems is further discussed and investigated. The coordinated control architecture proposed by Komenda and van Schuppen (2008) and further studied in Komenda et al. (2010a,b) in the case of prefix-closed global specification languages is generalized to the case of non-prefix-closed global specification languages and, correspondingly, to the non-prefix-closed plant languages. The solvability of the coordinated supervisory control synthesis problem is characterized in terms that are coordination-control counterparts of the terms of controllable and closed sublanguages used in supervisory control theory of monolithic discrete-event systems.

This paper presents necessary and sufficient conditions ensuring that a given global specification language is exactly achievable in our coordination-control architecture as the marked language of the closed-loop coordinated system, and that its prefix closure is exactly achievable as the prefix-closed language of the resulting coordinated system. Whence the closed-loop coordinated system is nonblocking. This means that under those conditions there exists a supervisor for the coordinator and supervisors for the local plants combined with the coordinator such that the specification language is exactly achieved in a nonblocking way.

The paper is organized as follows. First, Section 2 recalls the basic framework of the theory of supervisory control for discrete-event systems. Then, Section 3 presents the basic concepts of the coordinated control architecture. Section 4 formulates the fundamental problem of the supervisory control synthesis for distributed discrete-event systems using a coordinator. Section 5 provides an example. Finally, Section 6 concludes the paper and discusses the future work.

2. PRELIMINARIES

In this paper, the supervisory control framework introduced by Ramadge and Wonham (1987) is used. In this framework, discrete-event systems are modeled as generators that are deterministic finite-state machines with partial transition functions, i.e., so called incomplete finite-state machines.

Let E be a finite set (of events). Then, the set E^* denotes the free monoid generated by E, where the unit of E^* (the empty word) is denoted by ε. A *language* L over E is a subset of E^*.

A *generator* G is a construct $G = (Q, E, f, q_0, Q_m)$, where Q is the finite set of *states*, E is the finite set of *events*, $f : Q \times E \to Q$ is the *partial transition function*, $q_0 \in Q$ is the *initial state*, and $Q_m \subseteq Q$ is the set of *marked states*. In the usual way, f can be extended to a function from $Q \times E^*$ to Q. The language *generated* by G is defined as the set $L(G) = \{s \in E^* \mid f(q_0, s) \in Q\}$, and the language *marked* by G as the set $L_m(G) = \{s \in E^* \mid f(q_0, s) \in Q_m\}$.

The prefix closure of a language $L \subseteq E^*$ is defined as the set $\overline{L} = \{w \in E^* \mid \exists u \in E^*, wu \in L\}$ of all prefixes of all words from L. The language L is *prefix-closed* if $L = \overline{L}$.

Note that according to the definition, the generated language $L(G)$ is always prefix-closed, while the marked language $L_m(G)$ is not in general.

Let E be an event set. A *controlled generator* is a structure (G, E_c, Γ), where G is a generator over E, $E_c \subseteq E$ is the subset of *controllable events*, $E_u = E \setminus E_c$ is the subset of *uncontrollable events*, and $\Gamma = \{\gamma \subseteq E \mid E_u \subseteq \gamma\}$ is a *set of control patterns*.

A *supervisor* for the controlled generator (G, E_c, Γ) is a map $S : L(G) \to \Gamma$.

The *closed-loop system* associated with the controlled generator (G, E_c, Γ) and the supervisor S is defined as the minimal language $L(S/G)$ which satisfies (i) $\varepsilon \in L(S/G)$, and (ii) $s \in L(S/G), a \in S(s), sa \in L(G) \Rightarrow sa \in L(S/G)$. Furthermore, we define $L_m(S/G) = L(S/G) \cap L_m(G)$.

Note that the supervisor disables transitions in G, but it never disables a transition with an uncontrollable event.

If the closed-loop system is nonblocking, i.e.,

$$\overline{L_m(S/G)} = L(S/G)\,,$$

then the supervisor S is called *nonblocking*.

Definition 1. Let $L \subseteq E^*$ be a prefix-closed language, and let $E_u \subseteq E$ be the set of uncontrollable events. A language $K \subseteq E^*$ is *controllable* with respect to L and E_u if

$$\overline{K}E_u \cap L \subseteq \overline{K}\,.$$

Definition 2. Let G be a generator and $\emptyset \neq K \subseteq L_m(G)$ be a language. Then, the language K is $L_m(G)$*-closed* if

$$K = \overline{K} \cap L_m(G)\,.$$

However, it is obvious that the inclusion $K \subseteq \overline{K} \cap L_m(G)$ always holds.

Given a specification language K, a major control objective of supervisory control theory is to find a supervisor S so that $L_m(S/G) = K$ and $L(S/G) = \overline{K}$. It is well known that such a supervisor exists if and only if (1) K is controllable with respect to $L(G)$ and the set of uncontrollable events, and (2) it is $L_m(G)$-closed. For uncontrollable specifications, controllable sublanguages are considered.

In this paper, $\sup \mathrm{C}(K, L, E_u)$ denotes the supremal controllable sublanguage of the language K with respect to the language L and the uncontrollable event set E_u. This language always exists and equals the union of all controllable sublanguages of K, see Wonham (2009).

A *natural projection* $P : E^* \to E_0^*$, for any $E_0 \subseteq E$, is a homomorphism defined so that $P(a) = \varepsilon$, for $a \in E \setminus E_0$, and $P(a) = a$, for $a \in E_0$. The *inverse image* of P is denoted by $P^{-1} : E_0^* \to 2^{E^*}$.

Given sets $E_1, E_2, E_3, E_4 \subseteq E$, we use the generic notation $P_{3\cap4}^{1+2}$ to denote the natural projection from $E_1 \cup E_2$ to $E_3 \cap E_4$. Let $E_u \subseteq E$ be the set of uncontrollable events, then $E_{i,u} = E_u \cap E_i$, $i = 1, 2, 3, 4$, denotes the sets of locally uncontrollable events.

The synchronous product of two languages $L_1 \subseteq E_1^*$ and $L_2 \subseteq E_2^*$ is defined by

$$L_1 \| L_2 = P_1^{-1}(L_1) \cap P_2^{-1}(L_2) \subseteq E^*\,,$$

where $P_i : E^* \to E_i^*$, $i = 1, 2$, are natural projections. This notion is also defined for generators, see Wonham (2009) for more details. For generators G_1 and G_2, $L(G_1 \| G_2) = L(G_1) \| L(G_2)$ and $L_m(G_1 \| G_2) = L_m(G_1) \| L_m(G_2)$.

Note that in the automata framework, the supervisor S can be seen as a function from the set of states to Γ, which allows a generator representation of the supervisor. Then, the closed-loop system is a synchronous product of the supervisor S and the plant G, i.e., $L(S/G) = L(S) \| L(G)$, because Γ can be interpreted as an active event set of a given state of the supervisor.

3. CONCEPTS

This section presents the basic concepts of coordinated supervisory control for discrete-event systems.

Definition 3. Consider generators G_1, G_2, G_k. We call the generators G_1 and G_2 *conditionally independent* given G_k if

$$E_r(G_1 \| G_2) \cap E_r(G_1) \cap E_r(G_2) \subseteq E_r(G_k)\,,$$

where $E_r(G)$ is the set of all reachable events in G, i.e., there is no simultaneous move in both G_1 and G_2 without the coordinator G_k being also involved.

Note that $E_r(G_1 \| G_2)$ says that we restrict shared events of G_1 and G_2 to those events that are really generated in $L(G_1 \| G_2)$.

The definition can naturally be extended to three or more generators, or to languages.

Definition 4. A language K is called *conditionally decomposable* with respect to event sets E_1, E_2, E_k if

$$K = P_{1+k}(K) \| P_{2+k}(K) \| P_k(K) \tag{1}$$

and

$$\overline{K} = \overline{P_{1+k}(K)} \| \overline{P_{2+k}(K)} \| \overline{P_k(K)}\,. \tag{2}$$

Languages L_1 and L_2 are *synchronously nonconflicting* if

$$\overline{L_1 \| L_2} = \overline{L_1} \| \overline{L_2}\,.$$

Note that (2) of Definition 4 implies that the projected languages $P_{1+k}(K)$, $P_{2+k}(K)$, and $P_k(K)$ are synchronously nonconflicting. In addition, if K satisfies (1), then K is conditionally decomposable if and only if the languages $P_{1+k}(K)$, $P_{2+k}(K)$, and $P_k(K)$ are synchronously nonconflicting.

The following example justifies this definition by showing that none of these two properties are related.

Example 5. Let $E_1 = \{a_1, b_1, a, b\}$, $E_2 = \{a_2, b_2, a, b\}$, $E_k = \{a, b\}$, and $K = \{a_1a_2a, a_2a_1a, b_1b_2b, b_2b_1b\}$. Then, $P_{1+k}(K) = \{a_1a, b_1b\}$, $P_{2+k}(K) = \{a_2a, b_2b\}$, $P_k(K) = \{a, b\}$, and $K = P_{1+k}(K)\|P_{2+k}(K)\|P_k(K)$. Notice that $a_1b_2 \notin \overline{K}$, whereas $a_1b_2 \in \overline{P_{1+k}(K)\|P_{2+k}(K)\|P_k(K)}$.

Conversely, consider $L = \{\varepsilon, ab, ba, abc, bac\} \subseteq \{a, b, c\}^*$ with event sets $E_1 = \{a, c\}$, $E_2 = \{b, c\}$, $E_k = \{c\}$. Then, $\overline{L} = \overline{P_{1+k}(L)\|P_{2+k}(L)\|P_k(L)} = P_{1+k}(L)\|P_{2+k}(L)\|P_k(L)$, and it is obvious that $L \neq \overline{L}$. ◁

Note also that prefix-closed languages are nonconflicting.

Lemma 6. Let L_1 and L_2 be two prefix-closed languages. Then, they are nonconflicting.

Proof. Let L_1 and L_2 be prefix-closed languages. As the synchronous product of two prefix-closed languages is prefix-closed, we have that $\overline{L_1\|L_2} = L_1\|L_2 = \overline{L_1}\|\overline{L_2}$. □

4. CONTROL SYNTHESIS

In this section, we first formulate the coordinated control synthesis problem and then present necessary and sufficient conditions under which the problem can be solved.

Problem 7. Consider generators G_1, G_2, G_k over event sets E_1, E_2, E_k, respectively, and let $K \subseteq L_m(G_1\|G_2\|G_k)$ be a specification language. Assume that

(1) the coordinator G_k makes the generators G_1 and G_2 conditionally independent, and
(2) the specification language K is conditionally decomposable with respect to E_1, E_2, E_k.

Note that Condition (2) implies that $K \subseteq L_m(G_1\|G_2\|G_k)$ holds true if and only if $P_{1+k}(K) \subseteq L_m(G_1\|G_k)$ and $P_{2+k}(K) \subseteq L_m(G_2\|G_k)$. Thus, the inclusion can be checked without computing the whole system $G_1\|G_2$.

The coordinator supervisor S_k takes care of $P_k(K)$, i.e.,

$$L(S_k/G_k) \subseteq \overline{P_k(K)} \text{ and } L_m(S_k/G_k) \subseteq P_k(K)\,.$$

Similarly, local supervisors S_i take care of $P_{i+k}(K)$, $i = 1, 2$, i.e.,

$$L(S_i/[G_i\|(S_k/G_k)]) \subseteq \overline{P_{i+k}(K)} \text{ and } L_m(S_i/[G_i\|(S_k/G_k)]) \subseteq P_{i+k}(K).$$

The aim is to determine supervisors S_1, S_2, S_k for the respective generators so that the closed-loop system with the coordinator satisfies

$$L(S_1/[G_1\|(S_k/G_k)]) \parallel L(S_2/[G_2\|(S_k/G_k)]) \parallel L(S_k/G_k) = \overline{K} \quad (3)$$

and

$$L_m(S_1/[G_1\|(S_k/G_k)]) \parallel L_m(S_2/[G_2\|(S_k/G_k)]) \parallel L_m(S_k/G_k) = K\,. \quad (4)$$

◇

Note that if there exist such supervisors S_1, S_2, S_k, then their synchronous product is a nonblocking supervisor of the global plant.

Lemma 8. Consider the setting of Problem 7. If there exist supervisors S_1, S_2, S_k, then $S_1\|S_2\|S_k$ is a nonblocking supervisor of the global plant.

Proof. Let S_1, S_2, S_k be such supervisors. Then $\overline{K}$ can be written as

$$\begin{aligned}
&\overline{L_m((S_1/[G_1\|(S_k/G_k)])\|(S_2/[G_2\|(S_k/G_k)])\|(S_k/G_k))} \\
&= \overline{L_m(S_1\|G_1\|S_k\|G_k\|S_2\|G_2\|S_k\|G_k\|S_k\|G_k)} \\
&= \overline{L_m(S_1\|S_2\|S_k\|G_1\|G_2\|G_k)} \\
&= \overline{L_m([S_1\|S_2\|S_k]/[G_1\|G_2\|G_k])}\,.
\end{aligned}$$

As the formula from (3) can be reformulated in the same way, we obtain that

$$\overline{L_m([S_1\|S_2\|S_k]/[G_1\|G_2\|G_k])} = L([S_1\|S_2\|S_k]/[G_1\|G_2\|G_k])\,.$$

□

4.1 Conditional controllability

Conditional controllability is introduced by Komenda and van Schuppen (2008) and later studied by Komenda et al. (2010a,b) for prefix-closed global specification languages. In this paper, we discuss the generalization of this approach to the case of non-prefix-closed languages.

Definition 9. Consider the setting of Problem 7. Call the specification language $K \subseteq E^*$ *conditionally controllable* for generators (G_1, G_2, G_k) and sets $(E_{1+k,u}, E_{2+k,u}, E_{k,u})$ of uncontrollable events if

(1) the language $P_k(K) \subseteq E_k^*$ is controllable with respect to $L(G_k)$ and $E_{k,u}$;
(2) the language $P_{1+k}(K) \subseteq (E_1 \cup E_k)^*$ is controllable with respect to $L(G_1)\|\overline{P_k(K)}\|P_k^{2+k}(L(G_2)\|\overline{P_k(K)})$ and $E_{1+k,u} = E_u \cap (E_1 \cup E_k)$;
(3) the language $P_{2+k}(K) \subseteq (E_2 \cup E_k)^*$ is controllable with respect to $L(G_2)\|\overline{P_k(K)}\|P_k^{1+k}(L(G_1)\|\overline{P_k(K)})$ and $E_{2+k,u}$.

Now, we demonstrate that every conditionally-controllable and conditionally-decomposable language is also controllable.

Proposition 10. For $i = 1, 2, k$, let G_i be generators over E_i, $G = G_1\|G_2\|G_k$ and $E = E_1 \cup E_2 \cup E_k$. Let $K \subseteq L_m(G)$ be a conditionally-decomposable and conditionally-controllable language for the generators (G_1, G_2, G_k) and sets $(E_{1+k,u}, E_{2+k,u}, E_{k,u})$ of uncontrollable events. Then, the language K is controllable with respect to $L(G)$ and E_u.

To prove this proposition, the following auxiliary results are required. The first lemma is Proposition 4.6 in Feng (2007).

Lemma 11. (Controllability of the synchronous product). For $i = 1, 2$, let $L_i \subseteq E_i^*$ be prefix-closed languages and $K_i \subseteq L_i$ be controllable with respect to L_i and $E_{i,u}$, $E = E_1 \cup E_2$. If K_1 and K_2 are synchronously nonconflicting, then $K_1\|K_2$ is controllable with respect to $L_1\|L_2$ and E_u.

Lemma 12. (Transitivity of controllability). Let $K \subseteq L \subseteq M$ be languages over an event set E such that K is controllable with respect to $\overline{L}$ and E_u, and L is controllable with respect to $\overline{M}$ and E_u. Then, K is controllable with respect to $\overline{M}$ and E_u.

Proof. By assumptions, $\overline{K}E_u \cap \overline{L} \subseteq \overline{K}$ and $\overline{L}E_u \cap \overline{M} \subseteq \overline{L}$, and we want to show that $\overline{K}E_u \cap \overline{M} \subseteq \overline{K}$. Let $s \in \overline{K}$,

$a \in E_u$, and $sa \in \overline{M}$. Then, $K \subseteq L$ implies that $s \in \overline{L}$, and controllability of L implies that $sa \in \overline{L}$. However, $sa \in \overline{L}$ and controllability of K imply that $sa \in \overline{K}$. The proof is complete. □

Lemma 13. Let $L \subseteq E^*$ be a language and $P_k : E^* \to E_k^*$ with $E_k \subseteq E$ be a natural projection. Then, $L \| P_k(L) = L$.

Proof. By definition, $L \| P_k(L) = L \cap P_k^{-1} P_k(L) = L$. □

Proof of Proposition 10. Since the languages $P_k(K)$, $P_{1+k}(K)$, $P_{2+k}(K)$ are controllable with respect to $L(G_k)$ and $E_{k,u}$, $L(G_1) \| \overline{P_k(K)} \| P_k^{2+k}(L(G_2) \| \overline{P_k(K)})$ and $E_{1+k,u}$, and $L(G_2) \| \overline{P_k(K)} \| P_k^{1+k}(L(G_1) \| \overline{P_k(K)})$ and $E_{2+k,u}$, respectively, and by the remark following Definition 4 they are synchronously nonconflicting, Lemma 11 implies that $K = P_k(K) \| P_{1+k}(K) \| P_{2+k}(K)$ is controllable with respect to the synchronous product

$$\begin{aligned} L(G_k) \parallel\ & L(G_1) \| \overline{P_k(K)} \| P_k^{2+k}(L(G_2) \| \overline{P_k(K)}) \\ \parallel\ & L(G_2) \| \overline{P_k(K)} \| P_k^{1+k}(L(G_1) \| \overline{P_k(K)}) \\ = {} & L(G_k) \| L(G_1) \| L(G_2) \| \overline{P_k(K)} = L(G) \| \overline{P_k(K)} \end{aligned}$$

and E_u, where the equality is by commutativity of the synchronous product and Lemma 13.

As $P_k(K)$ is controllable with respect to $L(G_k)$ and $E_{k,u}$ by Definition 9, so is $\overline{P_k(K)}$. Thus, $L(G) \| \overline{P_k(K)}$ is controllable with respect to $L(G) \| L(G_k) = L(G)$ because the languages $L(G)$ and $\overline{P_k(K)}$ are synchronously nonconflicting (they are prefix-closed). By Lemma 12 and $K \subseteq L(G) \| \overline{P_k(K)} \subseteq L(G)$, K is controllable with respect to $L(G)$ and E_u. □

On the other hand, an example showing that the opposite inclusion does not hold can be found. It means that if K is controllable with respect to $L(G)$ and E_u, then it is not necessarily conditionally controllable. This holds true even if K is conditionally decomposable (see Komenda et al. (2010b)).

4.2 Conditionally-closed languages

Analogously to the notion of $L_m(G)$-closed languages, we define the notion of conditionally-closed languages.

Definition 14. Consider the setting of Problem 7. Call the specification language $\emptyset \neq K \subseteq E^*$ *conditionally closed* for generators (G_1, G_2, G_k) if

(1) the language $P_k(K) \subseteq E_k^*$ is $L_m(G_k)$-closed;
(2) the language $P_{1+k}(K) \subseteq (E_1 \cup E_k)^*$ is $\big(L_m(G_1) \| P_k(K)\big) \| P_k^{2+k}(L_m(G_2) \| P_k(K))$-closed;
(3) the language $P_{2+k}(K) \subseteq (E_2 \cup E_k)^*$ is $\big(L_m(G_2) \| P_k(K)\big) \| P_k^{1+k}(L_m(G_1) \| P_k(K))$-closed.

Note that if K is conditionally closed and conditionally controllable, there exists a supervisor S_k such that $L(S_k/G_k) = \overline{P_k(K)}$ and $L_m(S_k/G_k) = P_k(K)$, which is according to the basic theorem of supervisory control applied to $P_k(K)$ and S_k, cf. items (1) of Definitions 9 and 14.

As noted in (Cassandras and Lafortune, 2008, page 164), we can assume that the specification $K \subseteq L_m(G) \subseteq E^*$ is $L_m(G)$-closed. Then, so is the supremal controllable sublanguage $\sup \mathrm{C}(K, L(G), E_u)$. We show that it does not imply that $P_k(K)$ is $L_m(G_k)$-closed, for $G = G_1 \| G_2 \| G_k$ such that G_k makes G_1 and G_2 conditionally independent.

Example 15. Let $E = E_1 \cup E_2 = \{a_1, a\} \cup \{a_2, a\}$, and $E_k = \{a\}$. Define the language $K = \{a_1 a_2 a, a_2 a_1 a\}$. Then, $P_{1+k}(K) = \{a_1 a\}$, $P_{2+k}(K) = \{a_2 a\}$, $P_k(K) = \{a\}$, and it is not hard to verify that $K = P_{1+k}(K) \| P_{2+k}(K) \| P_k(K)$. Define the generators G_1, G_2, and G_k so that $L_m(G_1) = P_{1+k}(K)$, $L_m(G_2) = P_{2+k}(K)$, and $L_m(G_k) = \overline{P_k(K)} = \{\varepsilon, a\}$. Then, $L_m(G) = K$ and K is $L_m(G)$-closed. However, $P_k(K) \subset \overline{P_k(K)}$ is not $L_m(G_k)$-closed. ◁

This example demonstrates the pathological case, where $(L_m(G_k) \setminus P_k(K)) \cap \overline{P_k(K)} \neq \emptyset$, as shown below.

Proposition 16. Let G_i be generators over E_i, $i = 1, 2, k$, and let $G = G_1 \| G_2 \| G_k$. Let $K \subseteq L_m(G)$ be a language that is conditionally decomposable and conditionally closed for (G_1, G_2, G_k). Then, the language K is $L_m(G)$-closed if and only if

$$P_k^{-1}\big([L_m(G_k) \setminus P_k(K)] \cap \overline{P_k(K)}\big) \cap \big(\overline{K} \| L_m(G)\big) = \emptyset\,. \tag{5}$$

Proof. (⇒) Assume that $K = \overline{K} \cap L_m(G)$. As $\overline{K} \cap L_m(G) = \overline{K} \| L_m(G)$, $s \in \overline{K} \| L_m(G)$ implies that $P_k(s) \in P_k(K)$. Thus, $P_k(s) \notin L_m(G_k) \setminus P_k(K)$. The implication holds.

(⇐) To prove the opposite implication, assume that (5) holds, and that there exists a string $s \in \overline{K} \cap L_m(G)$ such that $s \notin K$. Then, $P_k(s) \in \overline{P_k(K)} \cap L_m(G_k)$. As the language K is conditionally decomposable and conditionally closed,

$$\begin{aligned} K &= P_k(K) \| P_{1+k}(K) \| P_{2+k}(K) \\ &= \overline{P_k(K)} \cap L_m(G_k) \\ &\quad \| \overline{P_{1+k}(K)} \cap L_m(G_1) \| P_k(K) \| P_k^{2+k}(L_m(G_2) \| P_k(K)) \\ &\quad \| \overline{P_{2+k}(K)} \cap L_m(G_2) \| P_k(K) \| P_k^{1+k}(L_m(G_1) \| P_k(K)), \\ &\qquad\qquad \text{by conditional closedness,} \\ &= \overline{P_k(K)} \| L_m(G_k) \\ &\quad \| \overline{P_{1+k}(K)} \| L_m(G_1) \| P_k(K) \| P_k^{2+k}(L_m(G_2) \| P_k(K)) \\ &\quad \| \overline{P_{2+k}(K)} \| L_m(G_2) \| P_k(K) \| P_k^{1+k}(L_m(G_1) \| P_k(K)), \\ &\qquad\qquad \text{by definition of synch. product,} \\ &= \overline{P_k(K)} \| L_m(G_k) \\ &\quad \| \overline{P_{1+k}(K)} \| L_m(G_1) \| P_k(K) \\ &\quad \| \overline{P_{2+k}(K)} \| L_m(G_2) \| P_k(K), \qquad \text{by Lemma 13,} \\ &= \overline{P_k(K)} \| \overline{P_{1+k}(K)} \| \overline{P_{2+k}(K)} \| P_k(K) \\ &\quad \| L_m(G_k) \| L_m(G_1) \| L_m(G_2), \qquad \text{by associativity,} \\ &= \overline{K} \| L_m(G) \| P_k(K) = \overline{K} \cap L_m(G) \cap P_k^{-1} P_k(K), \\ &\qquad\qquad \text{by conditional decomposability.} \end{aligned}$$

As $s \in \overline{K} \cap L_m(G)$, it is that $s \notin P_k^{-1} P_k(K)$. This means that $P_k(s) \notin P_k(K)$, which is a contradiction with (5). □

The following corollary is an immediate consequence of the previous result.

Corollary 17. Let G_i be generators over E_i, $i = 1, 2, k$, and let $G = G_1 \| G_2 \| G_k$. Let $K \subseteq L_m(G)$ be a language that is conditionally decomposable and conditionally closed for

(G_1, G_2, G_k). If $P_k(K)$ is prefix-closed, then K is $L_m(G)$-closed.

4.3 Control synthesis of conditionally-controllable and conditionally-closed languages

The following theorem presents the necessary and sufficient condition on a specification language to be exactly achievable in the coordinated control architecture.

Theorem 18. Consider the setting of Problem 7. There exist supervisors S_1, S_2, S_k such that

$$\begin{aligned} L(S_1/[G_1\|(S_k/G_k)]) \parallel L(S_2/[G_2\|(S_k/G_k)]) \\ \parallel L(S_k/G_k) \;=\; \overline{K} \\ \text{and} \\ L_m(S_1/[G_1\|(S_k/G_k)]) \parallel L_m(S_2/[G_2\|(S_k/G_k)]) \\ \parallel L_m(S_k/G_k) \;=\; K \end{aligned} \tag{6}$$

if and only if the specification language K is

(1) conditionally controllable with respect to generators (G_1, G_2, G_k) and sets $(E_{1+k,u}, E_{2+k,u}, E_{k,u})$ of uncontrollable events, and
(2) conditionally closed with respect to (G_1, G_2, G_k).

The interest in Theorem 18 is in the computational saving of the nonblocking supervisor. The distributed way of constructing supervisors S_k, S_1, S_2 is less complex than the construction of the global supervisor for the global plant $G_1\|G_2\|G_k$.

Note that the following conditions $L(S_k/G_k) \subseteq \overline{P_k(K)}$ and $L_m(S_k/G_k) \subseteq P_k(K)$ are required in Problem 7. Similarly, the conditions $L(S_i/[G_i\|(S_k/G_k)]) \subseteq \overline{P_{i+k}(K)}$ and their marked variants $L_m(S_i/[G_i\|(S_k/G_k)]) \subseteq P_{i+k}(K)$, for $i = 1, 2$, are required in Problem 7. Otherwise stated, we look for necessary conditions on global specification languages to have the maximal permissivity of the language resulting in our control scheme only in the reasonable case where safety can be achieved by the supervisors S_k, S_1, and S_2. We have proven that in such a case the properties of conditionally-controllable and conditionally-closed languages are necessary for the optimality. On the other hand, it has been proven in Komenda and van Schuppen (2008) that for the sufficiency we need not assume the inclusions above.

In practice, however, it is more interesting to know when safety holds when applying the control scheme combining a coordinator with local supervisors. Similarly as in the monolithic case it may happen that the maximal acceptable behavior given by the specification language K is not achievable using the coordination control scheme. It follows from Theorem 18 that such a situation occurs whenever the specification language K is either not conditionally controllable or not conditionally closed. In such a situation, a natural approach is to find the best approximation from below, i.e., to compute the supremal conditionally-controllable and conditionally-closed sublanguage if it exists. The reader is referred to the conclusion of this paper for more discussion concerning this problem.

5. EXAMPLE

In this section, we demonstrate the previous result in a simple example. Consider the following generators

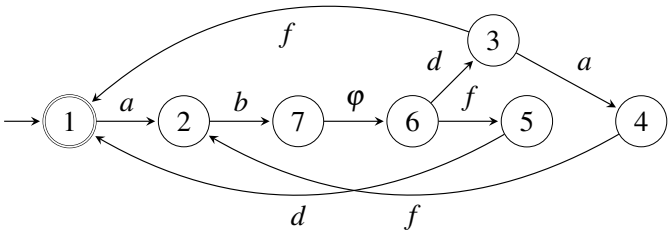


Fig. 1. The generator D.

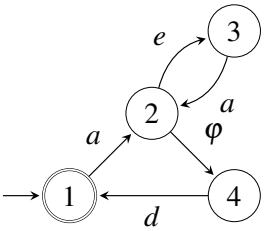
given over the event sets $E_1 = \{a, d, e, \varphi\}$ and $E_2 = \{b, f, \varphi\}$, respectively, where the set of controllable events is $E_c = \{e, b, \varphi\}$. The generator G_1 is defined as $G_1 = (\{1, 2, 3, 4\}, \{a, d, e, \varphi\}, f_1, 1, \{1\})$ with the transition function f_1 defined as in Fig. 2(a), and the generator G_2 as $G_2 = (\{1, 2, 3\}, \{b, \varphi, f\}, f_2, 1, \{1\})$ with the transition function f_2 defined as in Fig. 2(b). Assume that the specification language K is described by the following generator $D = (\{1, 2, 3, 4, 5, 6, 7\}, \{a, b, d, f, \varphi\}, \delta, 1, \{1\})$, where the transition function δ is defined as in Fig. 1.

Now, we briefly discuss one possible way how to find a coordinator so that the following two properties are satisfied: (i) G_k makes the two generators G_1 and G_2 conditionally independent, and (ii) the specification language K is conditionally decomposable with respect to the given event sets. We first need to find the coordinator alphabet E_k. By Definition 3, E_k must contain all shared events, i.e., E_k contains φ. This ensures that Condition (i) is satisfied. To satisfy Condition (ii), we need to extend the alphabet E_k. An efficient algorithm for finding such an extension is a part of our current research. However, it can be verified that for the choice $E_k = \{a, b, e, \varphi\}$, the condition is satisfied. Then, the coordinator G_k is chosen as a projection $P_k(G_1)\|P_k(G_2)$, i.e., $G_k = (\{1, 2, 3\}, \{a, b, e, \varphi\}, f_k, 1, \{1\})$ with the transition function f_k defined as in Fig. 2(c).

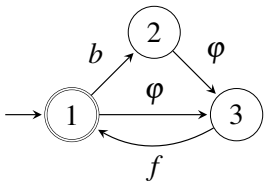
In addition, the reader can verify that the projected languages $P_k(K)$, $P_{1+k}(K)$, $P_{2+k}(K)$ are controllable with respect to $L(G_k)$, $L(G_1)\|\overline{P_k(K)}\|P_k^{2+k}(L(G_2)\|\overline{P_k(K)})$, and $L(G_2)\|\overline{P_k(K)}\|P_k^{1+k}(L(G_1)\|\overline{P_k(K)})$, respectively. This observation implies that the specification language K is conditionally controllable for generators (G_1, G_2, G_k) and the corresponding sets of locally uncontrollable events. Analogously, the reader can verify that the projected language $P_k(K)$ is $L_m(G_k)$-closed, the language $P_{1+k}(K)$ is $L_m(G_1)\|P_k(K)\|P_k^{2+k}(L_m(G_2)\|P_k(K))$-closed, and that $P_{2+k}(K)$ is $L_m(G_2)\|P_k(K)\|P_k^{1+k}(L_m(G_1)\|P_k(K))$-closed language. Thus, this observation implies that the specification language K is also conditionally closed for generators (G_1, G_2, G_k). According to the statement of Theorem 18, it immediately follows that there exist supervisors S_1, S_2, S_k such that the equations from the theorem holds. More specifically, the automata representations of supervisors S_1, S_2, and S_k coincide with the generators for languages $P_k(K)$, $P_{1+k}(K)$, and $P_{2+k}(K)$, respectively, depicted in Fig. 3. Finally, it is not hard to see that

$$\begin{aligned} L(S_1/[G_1\|(S_k/G_k)]) \parallel L(S_2/[G_2\|(S_k/G_k)]) \\ \parallel L(S_k/G_k)) \;=\; \overline{K} \end{aligned}$$

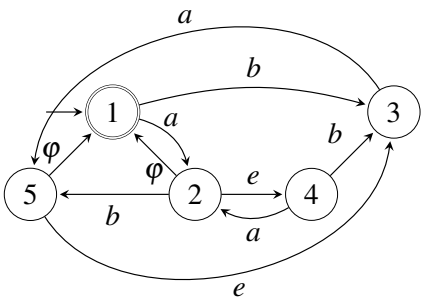
and



(a) The generator G_1.

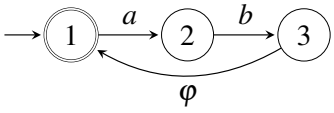


(b) The generator G_2.

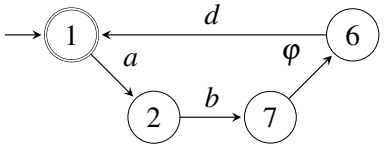


(c) The generator G_k.

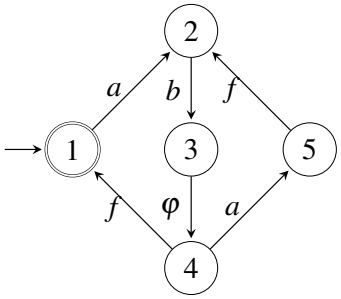
Fig. 2. Generators G_1, G_2, and G_k.



(a) Supervisor S_k generating $P_k(K)$.



(b) Supervisor S_1 generating $P_{1+k}(K)$.



(c) Supervisor S_2 generating $P_{2+k}(K)$.

Fig. 3. Supervisors S_k, S_1, and S_2.

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

which was to be shown.

6. CONCLUSION

In Komenda et al. (2010a,b), the case of prefix-closed global specification languages has been discussed. In those papers, the necessary and sufficient conditions placed on the global specification language to ensure the existence of a solution for the problem under consideration have been presented, and, in addition, for the specification languages that do not satisfy these conditions, a distributed procedure for the computation of the supremal sublanguage satisfying those conditions has been proposed.

In the present paper, this research has continued and the discussion has been generalized to the case of non-prefix-closed global specification languages and non-prefix-closed plant languages. The necessary and sufficient conditions for the existence of a solution for the problem have been proposed and discussed. However, in contrast to the prefix-closed specification languages, an analogous distributed procedure for the computation of the supremal sublanguage satisfying the required conditions for non-prefix-closed specification languages such that it is still conditionally decomposable (provided in the above references for prefix-closed languages) has not been derived yet since it turns out to be more complicated than in the prefix-closed case. Therefore, to construct such a procedure requires further investigation and is planned as a part of the future research.

ACKNOWLEDGEMENTS

The research received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant no. INFSO-ICT-224498, from the Czech Academy of Sciences, Institutional Research Plan no. AV0Z10190503, and from the GAČR grants no. 103/11/0517 and 202/11/P028.

REFERENCES

Cassandras, C.G. and Lafortune, S. (2008). *Introduction to discrete event systems, Second edition.* Springer.

Feng, L. (2007). *Computationally Efficient Supervisor Design for Discrete-Event Systems.* Ph.D. thesis, University of Toronto. URL `http://www.kth.se/polopoly_fs/1.24026!thesis.zip`.

Komenda, J., Masopust, T., and van Schuppen, J.H. (2010a). Supervisory control synthesis of discrete-event systems using coordination scheme. *CoRR 1007.2707.* URL `http://arxiv.org/abs/1007.2707`.

Komenda, J., Masopust, T., and van Schuppen, J.H. (2010b). Synthesis of safe sublanguages satisfying global specification using coordination scheme for discrete-event systems. In *Proc. of WODES 2010*, 436–441.

Komenda, J. and van Schuppen, J.H. (2008). Coordination control of discrete event systems. In *Proc. of WODES 2008*, 9–15.

Ramadge, P.J. and Wonham, W.M. (1987). Supervisory control of a class of discrete event processes. *SIAM J. Control Optim.*, 25(1), 206–230. doi:10.1137/0325013.

Wonham, W.M. (2009). Supervisory control of discrete-event systems. Lecture Notes, Department of Electrical and Computer Engineering, University of Toronto.