Synthesis of Safe Sublanguages satisfying Global Specification using Coordination Scheme for Discrete-Event Systems

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

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

Abstract: Modular control of discrete-event systems with a global specification and with only local supervisors is a difficult problem. For global specifications, however, the equivalent conditions may not be met. This paper formulates and solves a control synthesis problem for a generator with a global specification and with a combination of a coordinator and local controllers. Conditional controllability is proven to be an equivalent condition for the existence of such a coordinated controller. A procedure to compute a coordinated controller is provided.

Keywords: Discrete-event system, coordination control, coordinator, supervisory control, decentralized control, conditional controllability.

1. INTRODUCTION

In this paper supervisory control synthesis of modular discrete-event systems (DES) with a coordinator is investigated. DES represented as finite-state machines have been studied by P. J. Ramadge and W. M. Wonham (see e.g. Ramadge and Wonham (1989)). Large DES are typically formed as synchronous compositions of a large number of local components (subsystems) that are themselves finite state machines and run in parallel. Systems formed in this way are often called modular DES.

The aim of supervisory control is to ensure the control objectives of safety and of liveness to be satisfied by the closed-loop system. Typically, it is a so-called safety property, where the behavior (language) must be included in a specified language called specification. Since only so-called controllable specification languages can be achieved, one of the key issues in supervisory control synthesis is the computation of the supremal controllable sublanguage of the given specification language, from which the supervisor can be constructed.

The paper addresses control of distributed discrete-event systems. A distributed system, also called a modular system, consists of the interconnection of two or more subsystems. Then the global system refers to the parallel composition of all subsystems. There are two or more supervisors or controllers and the observable event sets of these supervisors are such that neither is contained in the other, the observations are thus completely incomparable. A distributed system differs from a system with decentralized control system in that the first is an interconnection of two or more subsystems and the latter has only one monolithic system. But both a distributed system and a decentralized system deal with two or more supervisors.

Control synthesis of a distributed system is difficult because the partial observations of the different supervisors differ. For technical or for economic reasons it is not possible or not preferred to send the partial observations of the different supervisors to one supervisor who then exerts global control. The specification is often global because it deals with the interactions of the subsystems.

In this paper the authors develop further the concept of coordination control proposed in Komenda and van Schuppen (2008), where a coordinator is applied for the control of modular discrete-event systems. The coordinator receives a part of events from the local subsystems and its task is to satisfy the global part of the specification and the nonblockingness. Hence, our coordinator can be seen as a two-way communication channel, where some events belonging to the coordinator event set are exchanged (communicated) between both subsystems. Coordination control may be seen as a reasonable trade-off between a purely decentralized control synthesis, which is in some cases unrealistic, and a global control synthesis, which is naturally prohibitive for complexity reasons. Moreover, our conditions obtained from the coordination control framework are based rather on the specification itself. In this paper we are only concerned with the safety issue. First, we propose a necessary and sufficient condition on a specification language to be exactly achieved in our coordination control architecture that consists of a coordinator, its supervisor, and the local supervisors for the subsystems. We call this condition *conditional controllability*, and it refines our condition that was only a sufficient one and has been presented in Komenda and van Schuppen (2008). It is shown that the supremal conditionally controllable sublanguage of a given specification always exists. In addition to the above mentioned existential result, a procedure for computation

of the supremal conditionally controllable sublanguage is proposed.

In the next section, decentralized supervisory control of modular DES is recalled and our coordination control approach is motivated. In Sections 2 and 3 we briefly recall the coordination control framework and concepts and in Section 4 our first result is presented: the equivalence condition on a specification language to be exactly achieved in our coordination control architecture. In Section 4 we show that the supremal conditionally controllable sublanguage always exists and in Section 5 a procedure for its computation is proposed. Finally, in Section 6, some concluding remarks are given including a discussion on future extensions of this work.

2. DECENTRALIZED AND COORDINATION CONTROL OF MODULAR DES

In this section, the elements of supervisory control theory (SCT) needed in the rest of this paper are recalled. We follow the standard framework of SCT, see the lecture notes Wonham (2009) or the book Cassandras and Lafortune (2008). DES are modeled as deterministic generators that are finite-state machines with partial transition functions. A (deterministic) *generator* G over a finite event set E is a structure $G = (Q, E, f, q_0, Q_m)$, where Q is a finite set of *states*, E is the finite *event set*, $f : Q \times E \to Q$ is the *partial transition function*, $q_0 \in Q$ is the *initial state*, and $Q_m \subseteq Q$ is the subset of *marked states*. If a transition labeled by $a \in E$ is defined for a state $q \in Q$, then this is denoted by $f(q, a)!$ Recall that f can be extended by induction to $f : Q \times E^* \to Q$ in the usual way. The behaviors of DES generators are defined in terms of languages. The *language* of G is defined as $L(G) = \{s \in E^* \mid f(q_0, s)!\}$, and the *marked language* of G is defined as $L_m(G) = \{s \in E^* \mid f(q_0, s) \in Q_m\}$.

The *natural projection* $P : E^* \to E_0^*$, where $E_0 \subseteq E$, is a homomorphism defined so that $P(a) = \epsilon$, 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 event sets E_i, E_j, and E_k, we use in this paper the notation P_k^{i+j} to denote the natural projection from $E_i \cup E_j$ to E_k, and $P_{j\cap k}^i$ to denote the natural projection from E_i to $E_j \cap E_k$.

Below, modular DES are considered. First, we recall that the synchronous product (also called the parallel composition) of 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 $P_i : E^* \to E_i^*$, for $i = 1, 2$, are natural projections to the local event sets. The synchronous product can also be defined for DES generators, the reader is referred to Cassandras and Lafortune (2008) for more details. In this case, it is well known that for two generators G_1 and G_2, $L(G_1 \| G_2) = L(G_1) \| L(G_2)$ and $L_m(G_1 \| G_2) = L_m(G_1) \| L_m(G_2)$.

A *controlled generator* (controlled DES, CDES) is a structure (G, E_c, Γ), where G is a generator, $E_c \subseteq E$ 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 called the *set of control patterns*. A *control supervisor* for the controlled generator (G, E_c, Γ) is a map $S : L(G) \to \Gamma$. The *closed-loop system* associated with a controlled generator (G, E_u, Γ) and a supervisor S is defined as the smallest language $L(S/G) \subseteq E^*$ which satisfies

(1) $\epsilon \in L(S/G)$,
(2) if $s \in L(S/G)$, $sa \in L(G)$, and $a \in S(s)$, then also $sa \in L(S/G)$.

Let us note that in the automata framework where the supervisor S is also represented by a DES generator, the closed-loop system can be recast as a synchronous product of the supervisor S and the plant G because it follows from the form of the control patterns that the supervisor never disables uncontrollable events, i.e., all uncontrollable transitions are always enabled. This is known as *admissibility* of a supervisor. Hence, for an admissible supervisor S that controls the plant G, one can write $L(S/G) = L(S) \| L(G)$.

We recall here that only controllable languages can be achieved by a supervisory controller as a closed-loop behavior, see Ramadge and Wonham (1989).

The prefix closure $\overline{L}$ of a language L is the set of all prefixes of all its words. A language $L \subseteq E^*$ is said to be prefix-closed if $L = \overline{L}$.

Definition 1. Let L be a prefix-closed language over an event set E with the uncontrollable event set $E_u \subseteq E$. A (specification) language $K \subseteq E^*$ is *controllable* with respect to L and E_u if

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

Given a prefix-closed specification language $K \subseteq E^*$, the goal of SCT is to find a supervisor S such that $L(S/G) = K$. It is known that such a supervisor exists if and only if K is controllable. Thus, for the specifications that are not controllable, controllable sublanguages of K are considered. The notation $\sup \mathrm{C}(K, L, E_u)$ is chosen for the supremal controllable sublanguage of K with respect to L and E_u. This language always exists and equals to the union of all controllable sublanguages of K, see e.g Cassandras and Lafortune (2008).

A modular DES is simply a synchronous product of two or more generators. Decentralized control synthesis of a modular DES is a procedure, where the control synthesis is carried out for each module or local subsystem. The global supervisor then formally consists of the synchronous product of the local supervisors though that product is not computed in practice. In terms of behaviors, the optimal global control synthesis is represented by the closed-loop language

$$\sup \mathrm{C}(K, L, E_u) = \sup \mathrm{C}(\|_{i=1}^n K_i, \|_{i=1}^n L_i, E_u) .$$

Given a rational global specification language $K \subseteq E^*$, one can theoretically always compute its supremal controllable sublanguage from which the optimal (least restrictive) supervisor can be built. Such a global control synthesis of a modular DES consists simply in computing the global plant and then the control synthesis is carried out as described above. However, the computational complexity of the global controller is for most practical control problems so high that other approaches have to be developed.

Decentralized control synthesis means that the specification language K is replaced by $K_i = K \cap P_i^{-1}(L_i)$, and the synthesis is done similarly as for the local specifications or using the notion of partial controllability, see Gaudin and Marchand (2004). Note the difference with decentralized control of monolithic plants as studied in Yoo and Lafortune (2002), where there are several control agents having different observations, but the system has no modular structure consisting of subsystems running in parallel.

However, the purely decentralized control synthesis is not always possible as the sufficient conditions under which it can be used are quite restrictive. Therefore, we have proposed coordination control in Komenda and van Schuppen (2008) as a trade-off between a purely decentralized control synthesis, which is in some cases unrealistic, and a global control synthesis, which is naturally prohibitive for complexity reasons.

3. CONCEPTS

Coordination control for DES is inspired by the concept of conditional independence of the theory of probability and of stochastic processes. Recall from Komenda and van Schuppen (2008) that conditional independence is roughly captured by the event set condition, when every joint action (move) of local subsystems must be accompanied by a coordinator action.

In this paper, after the architecture of our coordination scheme is recalled, a new necessary and sufficient condition on a specification language to be exactly achieved in this architecture is presented.

In the coordination scheme, first a supervisor S_k for the coordinator is synthesized that takes care of the part $P_k(K)$ of the specification language K. Then supervisors S_i, $i = 1, 2$, are synthesized so that the remaining parts of the specification, i.e., $P_{i+k}(K)$, are met by the new plant languages $G_i \| (S_k/G_k)$.

Definition 2. Consider three generators G_1, G_2, and G_k. We call G_1 and G_2 *conditionally independent* generators given G_k if in the global system there is no common transition of both G_1 and G_2 without the coordinator G_k being also involved. This condition can be written as

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

where $E_r(G)$ denotes the set of all reachable symbols in G, see also Komenda and van Schuppen (2008).

The concept is easily extended to the case of three or more generators. The corresponding concept in terms of languages follows.

Definition 3. Consider event sets E_1, E_2, E_k, and languages $L_1 \subseteq E_1^*$, $L_2 \subseteq E_2^*$, and $L_k \subseteq E_k^*$. The languages L_1 and L_2 are said to be *conditionally independent* given L_k if

$$E_r(L_1 \| L_2) \cap E_1 \cap E_2 \subseteq E_k\,,$$

where $E_r(L)$ denotes the set of all symbols occurring in words of L.

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

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

4. CONTROL SYNTHESIS OF CONDITIONALLY CONTROLLABLE LANGUAGES

Problem 5. Consider generators G_1, G_2, G_k with event sets E_1, E_2, E_k, respectively, and a prefix-closed specification language

$$K \subseteq L(G_1 \| G_2 \| G_k)\,.$$

It is assumed that K is prefix-closed because we only focus on the controllability issues in this paper, while nonblocking issues will be addressed in a future work. Assume that the coordinator G_k makes the two generators G_1 and G_2 conditionally independent, and that the specification language K is conditionally decomposable with respect to the event sets (E_1, E_2, E_k).

The overall control task is divided into local subtasks and the coordinator subtask, cf. Komenda and van Schuppen (2008). The coordinator takes care of its "part" of the specification, namely $P_k(K)$. Otherwise stated, S_k must be such that

$$L(S_k/G_k) \subseteq P_k(K)\,.$$

Similarly, supervisors S_1 and S_2 take care of their corresponding "parts" of the specification, namely $P_{i+k}(K)$, for $i = 1, 2$. Otherwise stated, S_i is such that for $i = 1, 2$,

$$L(S_i/[G_i \| (S_k/G_k)]) \subseteq P_{i+k}(K)\,,$$

Determine supervisors S_1, S_2, and S_k for the respective generators so that the closed-loop system with the coordinator is such that

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

The solution of Problem 5 requires the definition of the concept of conditional controllability which will be proven to be the characterization of the solution of that problem.

Let $E_u \subseteq E$ be the set of uncontrollable events and denote by $E_{i,u} = E_u \cap E_i$, for $i = 1, 2, k$, the corresponding sets of locally uncontrollable events. Moreover, let $E_{i+j} = E_i \cup E_j$, for $i, j \in \{1, 2, k\}$. Then $E_{i+j,u} = E_{i+j} \cap E_u$.

Definition 6. Consider the setting of Problem 5. Call the specification language $K \subseteq E^*$ *conditionally controllable* for generators (G_1, G_2, G_k) and for the (uncontrollable) event subsets $(E_{1+k,u}, E_{2+k,u}, E_{k,u})$ if

(i) The language $P_k(K) \subseteq E_k^*$ is controllable with respect to G_k and $E_{k,u}$; equivalently,

$$P_k(K) E_{k,u} \cap L(G_k) \subseteq P_k(K)\,.$$

Then there is a nonblocking supervisor S_k for G_k such that $L(S_k/G_k) = P_k(K)$. The supervisor S_k is used in the remaining part of the definition.

(ii.a) The language $P_{1+k}(K) \subseteq (E_1 \cup E_k)^*$ is controllable with respect to $L(G_1 \| (S_k/G_k)) \| P_k^{2+k} L(G_2 \| (S_k/G_k))$ and $E_{1+k,u} = E_u \cap (E_1 \cup E_k)$; equivalently,

$$P_{1+k}(K) E_{1+k,u} \cap L(G_1 \| (S_k/G_k)) \cap (P_k^{1+k})^{-1} P_k^{2+k} L(G_2 \| (S_k/G_k)) \subseteq P_{1+k}(K)\,.$$

(ii.b) The language $P_{2+k}(K) \subseteq (E_2 \cup E_k)^*$ is controllable with respect to $L(G_2 \| (S_k/G_k)) \| P_k^{1+k} L(G_1 \| (S_k/G_k))$ and $E_{2+k,u} = E_u \cap (E_2 \cup E_k)$; equivalently,

$$P_{2+k}(K)E_{2+k,u} \cap$$
$$L(G_2 \| (S_k/G_k)) \cap (P_k^{2+k})^{-1} P_k^{1+k} L(G_1 \| (S_k/G_k)) \subseteq P_{2+k}(K)\,.$$

The interpretation of the term after the intersection in (ii.a) is that the effect of Subsystem 2 in combination with the controlled coordinator system $G_2 \| (S_k/G_k)$ is taken into account when checking conditional controllability.

The conditions of Definition 6 can be checked by classical algorithms with low (polynomial) computational complexity for verification of the controllability as is directly clear from the definition. However, natural projections are involved. Still, if the corresponding satisfy observer property Wong and Wonham (1996), discussed and used further in the paper, then projected languages do not have larger representations than the original languages.

The computational complexity of checking conditional controllability is much less than that of controllability of the global system $L(G_1) \| L(G_2) \| L(G_k)$. This is because instead of checking the controllability with the global specification and the global system, we check it only on the corresponding projections to $E_k \cup E_1$ and $E_k \cup E_2$. The projections are smaller when they satisfy the observer property.

The following result from Wonham (2009) is useful.

Lemma 7. (Wonham (2009)). Let $P_k : E^* \to E_k^*$ be a natural projection, and let $L_1 \subseteq E_1^*$ and $L_2 \subseteq E_2^*$ be local languages over $E_1 \subseteq E$ and $E_2 \subseteq E$, respectively, such that $E_k \supseteq E_1 \cap E_2$. Then

$$P_k(L_1 \| L_2) = P_k(L_1) \| P_k(L_2)\,.$$

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

Theorem 8. Consider the setting of Problem 5 of control for safety. There are supervisors (S_1, S_2, S_k) such that

$$L(S_1/[G_1 \| (S_k/G_k)]) \parallel L(S_2/[G_2 \| (S_k/G_k)]) \parallel L(S_k/G_k)) = K \quad (1)$$

iff the specification K is conditionally controllable with respect to (G_1, G_2, G_k) and $(E_{1+k,u}, E_{2+k,u}, E_{k,u})$ of locally uncontrollable events.

The interest in Theorem 8 is in the computational saving of the computation of the supervisor, the distributed way of constructing successively the supervisors S_k, S_1, and S_2 is much less complex than the construction of the global supervisor for the system $G_1 \| G_2 \| G_k$.

Note that it is required that $L(S_k/G_k) \subseteq P_k(K)$. It follows from an example (see below) in our personal correspondence with Klaus Schmidt (U. Erlangen) that necessity in Theorem 8 cannot hold without this assumption. Similarly, $L(S_i/[G_i \| (S_k/G_k)]) \subseteq P_{i+k}(K)$, for $i = 1, 2$, is required. Otherwise stated, we are looking for necessary conditions on global specifications for having the maximal permissivity of the language resulting by the application of 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 conditional controllability is necessary for the optimality (maximal permissivity). It is clear from the proof that for the sufficiency part we need not assume the inclusions above (cf. Komenda and van Schuppen (2008)).

Example 9. (Schmidt (2008)). Consider the following DES generators $G_1 = (\{1,2,3,4\}, \{a,d,e,\varphi\}, f_1, 1, \{1\})$, where the set of controllable events is $\{a, \varphi\}$, and f_1 is defined as in Fig. 1, $G_2 = (\{1,2,3\}, \{b,\varphi,f\}, f_2, 1, \{1\})$, where

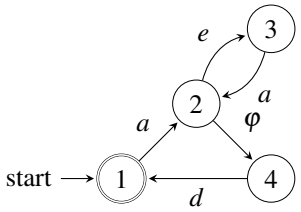

Fig. 1. The DES generator for G_1.

the set of controllable events is $\{b, \varphi\}$, and f_2 is defined as in Fig. 2, and $G_k = (\{1,2,3\}, \{a,b,\varphi\}, f_k, 1, \{1\})$,

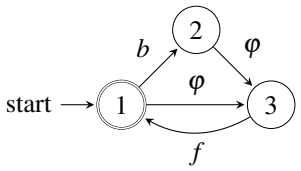

Fig. 2. The DES generator for G_2.

where the set of controllable events is $\{b, \varphi\}$, and f_k is defined as in Fig. 3. Assume that the specification

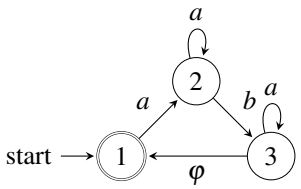

Fig. 3. The DES generator for G_k.

language K is described by the DES generator $D = (\{1,2,3,4,5,6,7\}, \{a,b,d,f,\varphi\}, \delta, 1, \{1\})$, where $\delta(1,a) = 2$, $\delta(2,b) = 7$, $\delta(7,\varphi) = 6$, $\delta(6,d) = 3$, $\delta(6,f) = 5$, $\delta(3,a) = 4$, $\delta(3,f) = 1$, $\delta(4,f) = 2$, and $\delta(5,d) = 1$.

It can be verified that G_k makes G_1 and G_2 conditionally independent and that the specification K is conditionally decomposable. In addition, $P_k(K)$ is not controllable with respect to $L(G_k)$, see Fig. 4 and 5. Analogously, it can be verified that $P_{2+k}(K)$ is not controllable with respect to $L(G_2 \| G_k) \| P_k^{1+k} L(G_1 \| G_k)$, and that $P_{1+k}(K)$ is controllable with respect to $L(G_1 \| G_k)$ and thus also with respect to $L(G_1 \| G_k) \| P_k^{2+k} L(G_2 \| G_k)$. However, it can be verified

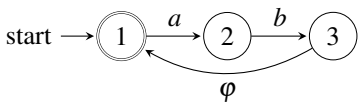

Fig. 4. The DES generator for $P_k(K)$.

that K is controllable with respect to $L(G_1 \| G_2 \| G_k)$ and a supervisory control to enforce K can be implemented by choosing the following supervisors.

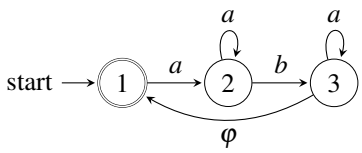

Fig. 5. The DES generator for $L(G_k)$.

- $S_k(s) = E_k = \{a, b, \varphi\}$ for all $s \in L(G_k)$, i.e., S_k enables all events; Nevertheless, note that it is not satisfied that $L(S_k/G_k) \subseteq P_k(K)$.
- $S_2(s) = E_2 \cup E_k = \{a, b, \varphi, f\}$ for all $s \in L(G_2)$, i.e., S_2 enables all events;
- S_1 is such that $L(S_1/[G_1 \| G_k]) = P_{1+k}(K)$.

Then

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

The problem is that coordinator G_k does not satisfy the assumption stated in Problem 5: There does not exist a supervisor S_k such that $L(S_k/G_k) \subseteq P_k(K)$ is nonempty. Note that in this example another coordinator can be chosen so that the setting of Problem 5 is matched, namely there exist a supervisor S_k with $L(S_k/G_k) \subseteq P_k(K)$ nonempty. It is sufficient to remove the selfloops in states 2 and 3 in Fig. 5.

In practice it is more interesting to know when safety (i.e., inclusion) holds when applying the overall 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 our coordination control scheme. It follows from Theorem 8 that in our case such a situation occurs whenever K is not conditionally controllable. A natural question is to find the best approximation from below: a conditionally controllable sublanguage. It turns out the following result holds true.

Theorem 10. The supremal conditionally controllable sublanguage of a given language K always exists and is equal to the union of all conditionally controllable sublanguages of K.

Proof. Similarly as for ordinary controllability it can be shown that conditionally controllability is preserved by language unions.

5. SUPREMAL CONDITIONALLY CONTROLLABLE SUBLANGUAGES

In what follows, we present a procedure for computation of the supremal conditionally controllable sublanguage.

Given generators G_1, G_2, and G_k, we denote $L_i = L(G_i)$, for $i = 1, 2, k$. Let $\sup \mathrm{CC}(K, L, (E_{1+k,u}, E_{2+k,u}, E_{k,u}))$ denote the supremal conditionally controllable sublanguage of K with respect to $L = L(G_1 \| G_2 \| G_k)$ and the sets of uncontrollable events $(E_{1+k,u}, E_{2+k,u}, E_{k,u})$.

Our approach is based on concepts from hierarchical supervisory control, which is natural, because our coordination control can be seen as a combination of decentralized and hierarchical supervisory control.

The following conditions of Wong and Wonham (1996) also discussed in Feng (2007) are required in the main result of this section. These conditions originate from the hierarchical supervisory control Wong and Wonham (1996). It should not be surprising that they play a role in our study, because coordination control can be seen as a particular instance of hierarchical control.

Definition 11. The natural projection $P_k : E^* \to E_k^*$, where $E_k \subseteq E$, is an *L-observer* for $L \subseteq E^*$ if, for all $t \in P(L)$ and $s \in \overline{L}$, $P(s) \leq t$ implies that there exists $u \in E^*$ such that $su \in L$ and $P(su) = t$.

Definition 12. The natural projection $P_k : E^* \to E_k^*$, where $E_k \subseteq E$, is *output control consistent* (OCC) for $L \subseteq E^*$ if for every $s \in \overline{L}$ of the form

$$s = \sigma_1 \dots \sigma_\ell \quad \text{or} \quad s = s'\sigma_0\sigma_1 \dots \sigma_\ell, \quad \ell \geq 1\,,$$

where $s' \in E^*$, $\sigma_0, \sigma_\ell \in E_k$ and $\sigma_i \in E \setminus E_k$, for $i = 1, 2, \dots, \ell-1$, if $\sigma_\ell \in E_u$, then $\sigma_i \in E_u$, for $i = 1, \dots, \ell-1$.

Now, we can now present the main result of this section, which gives a construction procedure for computation of the supremal conditionally controllable sublanguage.

Theorem 13. Let K and L be two prefix-closed languages over an event set E, and let the specification language K be conditionally decomposable. Define

$$\begin{aligned} \sup \mathrm{C}_k &= \sup \mathrm{C}(P_k(K) \| P_k(L_1 \| L_2) \| L_k, L_k, E_{k,u})\,, \\ \sup \mathrm{C}_{1+k} &= \sup \mathrm{C}(P_{1+k}(K) \| L_1, L_1 \| \sup \mathrm{C}_k, E_{1+k,u})\,, \\ \sup \mathrm{C}_{2+k} &= \sup \mathrm{C}(P_{2+k}(K) \| L_2, L_2 \| \sup \mathrm{C}_k, E_{2+k,u})\,. \end{aligned}$$

Let the projection P_k^{i+k} be an $(P_i^{i+k})^{-1}(L_i)$-observer and OCC for the language $(P_i^{i+k})^{-1}(L_i)$, for $i = 1, 2$. Then

$$\sup \mathrm{C}_k \| \sup \mathrm{C}_{1+k} \| \sup \mathrm{C}_{2+k} = \sup \mathrm{CC}(K \cap L, L, (E_{1+k,u}, E_{2+k,u}, E_{k,u}))\,.$$

Note that if we know that the specification language K is included in the global language L, the computation can be simplified as shown in the following corollary.

Corollary 14. Let $K \subseteq L$ be two prefix-closed languages over an event set E, and let K be conditionally decomposable. Define

$$\begin{aligned} \sup \mathrm{C}_k &= \sup \mathrm{C}(P_k(K), L_k, E_{k,u})\,, \\ \sup \mathrm{C}_{1+k} &= \sup \mathrm{C}(P_{1+k}(K), L_1 \| \sup \mathrm{C}_k, E_{1+k,u})\,, \\ \sup \mathrm{C}_{2+k} &= \sup \mathrm{C}(P_{2+k}(K), L_2 \| \sup \mathrm{C}_k, E_{2+k,u})\,. \end{aligned}$$

Let P_k^{i+k} be an $(P_i^{i+k})^{-1}(L_i)$-observer and OCC for $(P_i^{i+k})^{-1}(L_i)$, for $i = 1, 2$. Then

$$\sup \mathrm{C}_k \| \sup \mathrm{C}_{1+k} \| \sup \mathrm{C}_{2+k} = \sup \mathrm{CC}(K, L, (E_{1+k,u}, E_{2+k,u}, E_{k,u}))\,.$$

Proof. If $K \subseteq L$, then

$$\begin{aligned} P_k(K) \subseteq P_k(L) &= P_k(L_1 \| L_2 \| L_k) \\ &= P_k(L_1 \| L_2) \| L_k, \quad \text{by Lemma 7}\,. \end{aligned}$$

From $L_1 \| L_2 \| L_k = P_1^{-1}(L_1) \cap P_2^{-1}(L_2) \cap P_k^{-1}(L_k)$, we also have

$$\begin{aligned} P_{i+k}(K) &\subseteq P_{i+k}(P_i^{-1}(L_i)) \\ &= (P_i^{i+k})^{-1}(L_i)\,, \end{aligned}$$

for $i = 1, 2$. Since $P_k(K) \subseteq P_k(L_1 \| L_2) \| L_k$ and $P_{i+k}(K) \subseteq (P_i^{i+k})^{-1}(L_i)$, $i = 1, 2$, the proof follows from the previous theorem. □

In addition to a procedure for computation of $\sup \mathrm{CC}$ in a distributed way, another consequence of theorem above is interesting. Namely, under the conditions of Theorem 13 $\sup \mathrm{CC}$ is conditionally decomposable, cf. Lemma 15.

Lemma 15. A language $M \subseteq E^*$ is conditionally decomposable with respect to the event sets (E_1, E_2, E_k) iff there exist languages $M_i \subseteq E_i^*$, $i = 1, 2, k$, such that

$$M = M_1 \| M_2 \| M_k.$$

Proof. Conditionally decomposability of M means $M = P_1(M) \| P_2(M) \| P_k(M)$. Let $M_i = P_i(M)$, $i = 1, 2, k$. Then the necessity is proven. To prove sufficiency, assume there are languages $M_i \subseteq E_i^*$, $i = 1, 2, k$, such that $M = M_1 \| M_2 \| M_k$. Obviously, $P_i(M) \subseteq M_i$, $i = 1, 2, k$, which implies $P_k(M) \| P_1(M) \| P_2(M) \subseteq M$. As $M \subseteq P_i^{-1} P_i(M)$, $i = 1, 2, k$, and by the definition of synchronous product we also obtain $M \subseteq P_k(M) \| P_1(M) \| P_2(M)$. □

Even more, this implies that the supremal conditionally controllable sublanguage is controllable with respect to the global plant as we show below, and, consequently, the supremal conditionally controllable sublanguage is included in the global supremal controllable sublanguage. This is not a surprise, because the language synthesized using our coordination architecture is more restrictive than the language synthesized using (monolithic) supervisory control of global plant.

Theorem 16. In the setting of Theorem 13 above we have

$$\sup \mathrm{CC}(K, L, (E_{1+k,u}, E_{2+k,u}, E_{k,u})) \subseteq \sup \mathrm{C}(K, L, E_u).$$

Proof. It is sufficient to show that

$$\sup \mathrm{CC} := \sup \mathrm{CC}(K, L, (E_{1+k,u}, E_{2+k,u}, E_{k,u}))$$

is controllable with respect to $L = L_1 \| L_2 \| L_k$ and E_u. Notice that there exist $\sup \mathrm{C}_k \subseteq E_k^*$, $\sup \mathrm{C}_{1+k} \subseteq E_{1+k}^*$, and $\sup \mathrm{C}_{2+k} \subseteq E_{2+k}^*$ as defined in Theorem 13 (respectively in Corollary 14) so that

$$\sup \mathrm{CC} = \sup \mathrm{C}_k \| \sup \mathrm{C}_{1+k} \| \sup \mathrm{C}_{2+k}\,.$$

In addition, we know that

- $\sup \mathrm{C}_k$ is controllable with respect to L_k and $E_{k,u}$,
- $\sup \mathrm{C}_{1+k}$ is controllable wrt. $L_1 \| \sup \mathrm{C}_k$ and $E_{1+k,u}$,
- $\sup \mathrm{C}_{2+k}$ is controllable wrt. $L_2 \| \sup \mathrm{C}_k$ and $E_{2+k,u}$.

By Proposition 4.6 in Feng (2007) (since all the languages under consideration are prefix-closed)

$$\sup \mathrm{CC} = \sup \mathrm{C}_k \| \sup \mathrm{C}_{1+k} \| \sup \mathrm{C}_{2+k}$$

is controllable with respect to

$$L_k \| L_1 \| \sup \mathrm{C}_k \| L_2 \| \sup \mathrm{C}_k = L \| \sup \mathrm{C}_k$$

and E_u. Analogously, we can obtain that $L \| \sup \mathrm{C}_k$ is controllable with respect to $L \| L_k = L$ and E_u.

Finally, by transitivity of controllability for prefix-closed languages we obtain that $\sup \mathrm{CC}$ is controllable with respect to L and E_u, which was to be shown. □

6. CONCLUSION

Supervisory control of modular DES with global specifications has been considered. A coordination control framework has been adopted where, unlike the purely decentralized setting, a global layer with a coordinator acting on a subset of the global event set has been added. Two main results have been presented: A necessary and sufficient condition on a specification to be exactly achieved in our coordination control architecture, called conditional controllability, has been proposed, and it has been shown how the supremal conditionally controllable sublanguage can be synthesized.

In this paper we have been interested only in the optimality of our control scheme, but blocking that is inherent to modular and, more generally, to our coordinated control synthesis has not been considered. It was then sufficient to choose a suitable coordinator event set and the coordinator itself need not impose any restriction on the behavior because its supervisor can take care of a required restriction of the plant projected to the coordinator events.

In a future paper it is our plan to address the blocking issue by considering a suitable coordinator and combine it with the three supervisors so that both blocking and maximal permissiveness are handled within our coordination scheme.

Thus, more work on coordination control dealing with global specification languages is needed. In particular, the synthesis of coordinators for nonblockingness is to be developed and the approach should be extended to partially observed modular plants and to classes of timed automata.

ACKNOWLEDGEMENTS

A comment by Klaus Schmidt (U. Erlangen) is herewith gratefully acknowledged.

This work was supported by the EU.ICT 7FP project DISC no. 224498, and by the Czech Academy of Sciences, Institutional Research Plan no. AV0Z10190503.

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. http://www.kth.se/polopoly_fs/1.24026!thesis.zip.

Gaudin, B. and Marchand, H. (2004). Supervisory control of product and hierarchical discrete event systems. *Eur. J. Control*, 10(2), 131–145.

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. (1989). The control of discrete event systems. *Proc. of IEEE*, 77(1), 81–98.

Schmidt, K. (2008). Personal correspondence.

Wong, K.C. and Wonham, W.M. (1996). Hierarchical control of discrete-event systems. *Discrete Event Dyn. Syst.*, 6(3), 241–273.

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

Yoo, T. and Lafortune, S. (2002). A general architecture for decentralized supervisory control of discrete-event systems. *Discrete Event Dyn. Syst.*, 12(3), 335–377.