Bottom-Up Approach to Multilevel Supervisory Control with Coordination

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

Abstract— **A multilevel coordination approach is proposed to lower the complexity of control synthesis of large-scale discrete-event systems. The bottom-up control synthesis method requires only conditional decomposability and conditional controllability of the system and of the specification unlike the top-down approach that requires the specification to be conditionally decomposable and conditionally controllable with respect to the multilevel architecture. The computation of coordinators and supervisors on different levels is presented. An academic example of two level control architecture is provided.**

I. Introduction

Large discrete-event systems (DES) with synchronous communication are often modeled as a parallel composition of several subsystems, typically automata or Petri nets [1]. In the case of automata (finite-state machines), such automata networks are called modular DES. Supervisory control [7] of DES has been introduced as a formal approach to guarantee that the closed-loop system satisfies the control objectives of safety and of nonblockingness. The modular (sometimes also called decentralized) control synthesis then consists in synthesizing local nonblocking supervisors for each of the subsystems separately. There are many papers on modular control combined with hierarchical approach that are not listed due to space restrictions, but can be found in [5]. It is, however, well known that modular approaches fail in general to guarantee both nonblockingness and maximally permissive safe behavior. Therefore, coordination control has been proposed in [6] as a trade-off between the computationally cheap purely decentralized control synthesis and the computationally expensive global control synthesis. It relies on concepts of conditional decomposability and conditional controllability, which form necessary and sufficient conditions on a specification to be achieved by the coordination control architecture. Constructive results, namely a computation scheme for construction of maximally permissive local supervisors, have been presented in [5]. These results rely on important concepts of hierarchical supervisory control, such as the *observer property* of [9] and *local control consistency* (LCC) of [8].

We distinguish between coordinators for safety and coordinators for nonblockingness. A coordinator for safety has been defined in [5] as the modular plant projected on the coordinator alphabet. This choice guarantees that the coordinator itself does not affect the behavior of the plant, because the synchronous product of the plant with its projection yields the plant itself. The only important step of control design is to determine the coordinator alphabet so that the specification satisfies conditional controllability, which is a distributed version of controllability. Global safety specifications can then be imposed in the maximally permissive way, that is, the supremal conditionally controllable sublanguages can be computed in a distributed way. If necessary, a coordinator for nonblockingness has to be computed as described in [5].

The contribution of this paper is to propose for modular DES with a large number of local components a multilevel coordination setting, instead of earlier approaches based on a single central coordinator. In this approach, local automata are grouped (divided) into several groups of automata such that within each group there is only a very small number of shared events. This ensures that most of the coordination task is done at the lowest level. The computational scheme proposed in this paper is referred to as the bottom-up approach, because we start at the lowest level, where for each group of automata the corresponding part of the specification is imposed by using coordination control and the results are reused at the upper level, where the global specification has to be imposed again by applying coordination control synthesis. Unlike the dual approach, called top-down and studied in [4], the specification is not made a priori decomposable with respect to all levels (in the top-down way), but after moving up from the lowest level we have to compute other supervisors that ensure the safety. At each level of the hierarchy both coordinators for safety and for nonblockingness are computed for each group of subsystems.

The paper has the following structure. The next section presents auxiliary results from supervisory control including coordination control with one central coordinator. The problem is stated in Section III. Then, in Section IV, a multilevel hierarchical structure is described, and in Section V, a bottom-up approach to multilevel coordination control is presented. Conclusions together with hints on future developments are given in Section VII.

II. Supervisory Control

This paper is based on the supervisory control framework introduced by Ramadge and Wonham [7]. In this framework, discrete-event systems are modeled as generators that are deterministic finite-state machines with partial transition functions. For a finite set A of events, also called alphabet, the standard notation A^* is used to denote the free monoid of

J. Komenda and T. Masopust are with Institute of Mathematics, Academy of Sciences of the Czech Republic, Žižkova 22, 616 62 Brno, Czech Rep. komenda@math.cas.cz, masopust@math.cas.cz

Jan H. van Schuppen is with Van Schuppen Control Research, Gouden Leeuw 143, 1103 KB, Amsterdam, The Netherlands. jan.h.van.schuppen@xs4all.nl

words generated by A. The unit element of A^* is the empty word denoted by ε. A language L over A is a subset of A^*.

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

The prefix closure $\overline{L} = \{w \in A^* \mid \exists v \in A^* \text{ such that } wv \in L\}$ of a language $L \subseteq A^*$ is the set of all prefixes of all its words. A language L is called prefix-closed if $L = \overline{L}$. In particular, $L(G)$ is prefix-closed.

A *controlled generator* is a triple (G, A_c, Γ), where G is a generator over A, $A_c \subseteq A$ is a set of *controllable events*, $A_u = A \setminus A_c$ is the set of *uncontrollable events*, and $\Gamma = \{\gamma \subseteq A \mid A_u \subseteq \gamma\}$ is the *set of control patterns* that correpond to enabled events. A *supervisor* runs in parallel with G, but it never disables an uncontrollable transition. This results in the closed-loop language $L(S/G) = L(S) \| L(G)$. The marked closed-loop languageis $L_m(S/G) = L(S/G) \cap L_m(G)$

Let L_m and L be languages over an alphabet A, where L is prefix-closed. A language $K \subseteq A^*$ is *controllable* with respect to L and A_u if $\overline{K} A_u \cap L \subseteq \overline{K}$. We recall that K is L_m-*closed* if $K = \overline{K} \cap L_m$. These conditions are equivalent to the existence of a nonblocking supervisor S such that $L_m(S/G) = K$, see [1], [10].

Recall that the synchronous product of languages $L_i \subseteq A_i^*$, $i = 1, \ldots, n$, is defined by

$$\|_{i=1}^n L_i = \cap_{i=1}^n P_i^{-1}(L_i) \subseteq A^*,$$

where $A = \cup_{i=1}^n A_i$ and $P_i : A^* \to A_i^*$, for $i = 1, \ldots, n$, are projections to local alphabets. The synchronous product can also be defined in terms of generators (the reader is referred to [1] for more details). In this case, for generators G_i, $i = 1, \ldots, n$, it is well known that $L(\|_{i=1}^n G_i) = \|_{i=1}^n L(G_i)$ and $L_m(\|_{i=1}^n G_i) = \|_{i=1}^n L_m(G_i)$.

III. Problem and Example

A reader is assumed to be familiar with our earlier results from [3], [5] using coordination supervisory control framework with a single (central) coordinator that is recalled in Appendix I. A serious issue with the coordination control framework relying on a single (central) coordinator is that for a large number of subsystems the coordinator alphabet has to be too large in order to satisfy all assumptions imposed by our framework such as conditional decomposability, observer and LCC conditions.

It should be understood that the assumption of conditional decomposability is better suited for systems composed of a small number of components. With a higher number of components it is not interesting to communicate the coordinator event to all components, but only to those where it is necessary. On the other hand, it often happens that there are different groups of subsystems that require different coordination events, which is not allowed in the original version of conditional decomposability. As a simple motivating example let us consider the simple modular plant and global specification below.

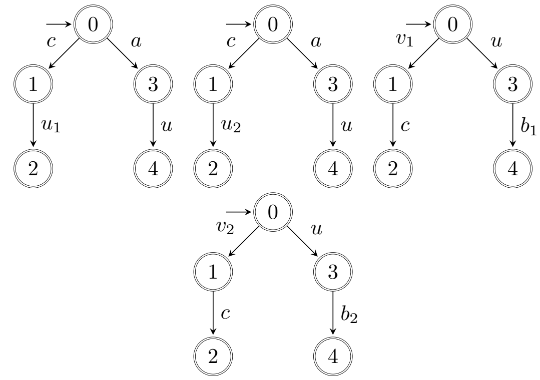

Fig. 1. Generators $G_1, \ldots, G_4$.

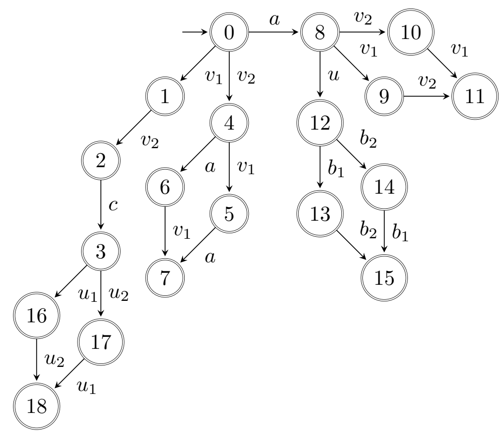

Fig. 2. Generator for the specification K.

Example 1: Let us consider four generators $G_1, \ldots, G_4$ over alphabets $A_1, \ldots, A_4$, as defined in Fig. 1, respectively, and their synchronous product $G = G_1 \| \ldots \| G_4$. Let $A_u = \{u, u_1, u_2\}$. The specification K is defined in Fig. 2.

It is easy to see that K is not decomposable wrt local alphabets A_i, $i = 1, 2, 3, 4$. A coordinator alphabet A_k needs to be found such that K is conditionally decomposable. However, this amounts to communicate the events from A_k among all subsystems.

On the other hand, it is easy to see that $K = P_{1+2}(K) \| P_{3+4}(K)$ and that $P_{1+2}(K) = P_1(K) \| P_2(K)$ is decomposable wrt alphabets A_1 and A_2. Even though $P_{3+4}(K)$ is not decomposable wrt alphabets A_3 and A_4, it is sufficient to include event v_1 in A_k in addition to shared events of G_3 and G_4, i.e., take $A_k = \{c, u, v_1\}$ and $P_{3+4}(K) = P_{3+k}(K) \| P_{4+k}(K)$ is actually conditionally decomposable wrt A_3, A_4, and A_k. It then appears natural that a multilevel coordination would be more advantageous for this example, because event v_1 needs to be communicated between A_3 and A_4 and not among all the four subsystems. ◁

In the example we have seen that it is then useless (and moreover it brings an unnecessary computational burden) to decompose a global specification according to the original definition of conditional decomposability with centralized coordination (see Appendix I), because it amounts to communicate all coordinator events among all local subsystems. For large systems with many components it is typically better to conditionally decompose a global specification according to the respective local alphabets in a more clever hierarchical way.

In the hierarchical structure of coordinators on several levels these conditions are only required for a given group of subsystems and because of this smaller extensions of the coordinator alphabet may be applied to make the observer and/or LCC conditions hold. Therefore, computationally efficient approaches to supervisory control are proposed in this section for large modular discrete-event systems modeled by synchronous products of automata.

IV. Multilevel Hierarchy of Subsystems and Coordinators

Let us consider the modular DES $G = G_1 \| G_2 \| \dots \| G_n$. We have proposed in [4] a technique for the organization of local automata into groups of automata on several levels. The guiding strategy was to gather local automata with strong interactions at the lowest level of the multilevel structure. A criterion for this division of automata into groups can be the number of shared events within the groups of subsystems.

Assume the organization of subsystems into groups is given with indexing of the generators changed so that the first group is formed by generators $G_1, \dots, G_{i_1}$, the second group is formed by $G_{i_1+1}, \dots, G_{i_2}$, and so forth, i.e., the m-th group is formed by $G_{i_{m-1}+1}, \dots, G_{i_m}$, where $1 \le i_1 \le i_2 \le \dots \le i_m = n$.

We denote the indices of the generators of the j-th group by I_j, i.e., $I_j = \{i_{j-1}+1, i_{j-1}+2, \dots, i_j\}$, $j = 1, \dots, m$ where $i_0 = 1$. Similarly, we may assume that the groups of subsystems $I_1, \dots, I_m$ are organized into l larger groups $J_1, \dots, J_\ell$ with $\ell \le m$ using the same criterion (but applied to groups rather than low-level automata themselves). In order to avoid too many indices we only consider the two-level organization in this paper, i.e., $J_1 = \{I_1, \dots, I_m\}$ meaning that $\|_{i \in I}^{I \in J_1} G_i = G_1 \| \dots \| G_n$.

The notation $A_{sh,j}$ is chosen for the set of shared events of generators $G_{i_{j-1}+1}, \dots, G_{i_j}$ of group I_j, i.e.,

$$A_{sh,j} = \bigcup_{k,\ell \in \{i_{j-1}+1, \dots, i_j\}}^{k \neq \ell} (A_k \cap A_\ell)\,.$$

Unlike the previously studied case with one central coordinator, there are m low-level coordinators $G_{k_1}, \dots, G_{k_m}$ at the low level, one for each group of subsystems. The situation is depicted in Fig. 3. Moreover, there is one high-level coordinator denoted by G_k.

The notation $A_{I_r} = \cup_{i \in I_r} A_i$ for the alphabet of low level groups of generators is used in the paper. P_{I_r} then denotes the projection $P_{I_r} : A^* \to A_{I_r}^*$. The high-level coordinator is over the alphabet A_k that is chosen in such a way that it contains all shared events, in this case all events shared by the groups of subsystems denoted by

$$A_{sh} = \bigcup_{k,\ell \in \{1, \dots, m\}}^{k \neq l} (A_{I_k} \cap A_{I_\ell})\,.$$

V. Control Synthesis – Bottom-Up Approach

In this section, a bottom-up approach to the coordination control synthesis is presented. Unlike the top-down approach discussed in another paper we start at the bottom (lowest) level. The advantage of this approach is that we do not need to assume that K is two-level conditional decomposable and that the standard notion of conditional decomposability as well as conditional controllability can be used. Let us recall that the r-th group on the local level is formed by $G_{i_{r-1}+1}$, $\dots$, G_{i_r}, where $1 \le r \le m$. We denote the composed language of the r-th group by $L(G_{I_r})$, i.e., $L(G_{I_r}) = L(\|_{\ell=i_{r-1}+1}^{i_r} G_\ell) = \|_{\ell=i_{r-1}+1}^{i_r} L(G_\ell)$.

The bottom-up approach consists of the following: the supervisory control task given by specification K is divided into subtasks $P_{I_r}(K)$, $r = 1, \dots, m$, that are treated separately. For each subtask, a standard coordination control with a single (central) coordinator is applied. This means that first $P_{I_r}(K)$ is made conditionally decomposable, for each $r = 1, \dots, m$, by finding the coordinator alphabets A_{k_r} so that

$$P_{I_r}(K) = \|_{j \in I_r} P_{j+k_r}(K)\,.$$

The corresponding coordinator for the r-th group of the subsystems is computed in the standard way as $G_{k_r} = P_{k_r}(\|_{\ell=i_{r-1}+1}^{i_r} G_\ell) = \|_{\ell \in I_r} P_{k_r}(G_\ell)$ by assuming that A_{k_r} has been chosen so that P_{k_r} are all L_i-observers, where $L_i = L(G_i)$, $i \in I_r$. For simplicity $L(G_{k_r})$ is denoted by L_{k_r} for $r = 1, \dots, m$. If $P_{I_r}(K)$ is not conditionally controllable (see Definition 6) for some $r = 1, \dots, m$, then the supremal conditionally controllable sublanguage of $P_{I_r}(K)$, recalled in the appendix and denoted by $\sup \mathrm{cC}(P_{I_r}(K)) = \sup \mathrm{cC}(P_{I_r}(K), (L_i)_{i \in I_r}, L_{k_r}, (A_{i,u})_{i \in I_r}, A_{k_r,u})$ has to be computed by the technique presented in Corollary 7. Namely, for group $I_j, j = 1, \dots, m$, we compute $\sup \mathrm{C}_k^j = \sup \mathrm{C}(P_{k_j}(K), L_{k_j}, A_{k_j,u})$ and for any $i \in I_j$, $\sup \mathrm{C}_{i+k}^j = \sup \mathrm{C}(P_{i+k_j}(K), L_i \| \sup \mathrm{C}_k^j, A_{i+k,u})$. Then under conditions of Corollary 7 (that can be imposed by extending coordinator alphabets $A_{k_j}, j = 1, \dots, m$)

$$\sup \mathrm{cC}(P_{I_j}(K)) = \|_{i \in I_j} \sup \mathrm{C}_{i+k}^j. \tag{1}$$

If there exists a $r \in \{1, \dots, m\}$ such that $P_{I_r}(K)$ is conditionally controllable, then $\sup \mathrm{cC}(P_{I_j}(K)) = P_{I_j}(K)$. Note that for $P_{I_j}(K)$ that are not prefix-closed the above computation yields $\sup \mathrm{cC}(P_{I_j}(K))$ only if $\sup \mathrm{C}_k \subseteq P_k(\sup \mathrm{C}_{i+k})$ for all $i \in I_j$, otherwise optimality is lost. Moreover, for $P_{I_j}(K)$ that are not prefix-closed it could well be that languages $\sup \mathrm{C}_{i+k}^j$, $i \in I_j$, are conflicting, which leads to blocking. In this case Theorem 9 can be applied (possibly after extending alphabets A_{k_j} such that observer conditions are satisfied) we define coordinators for nonblockingness as

$$C_{k_j} = \|_{i \in I_j} P_{k_j}(\sup \mathrm{C}_{i+k}^j). \tag{2}$$

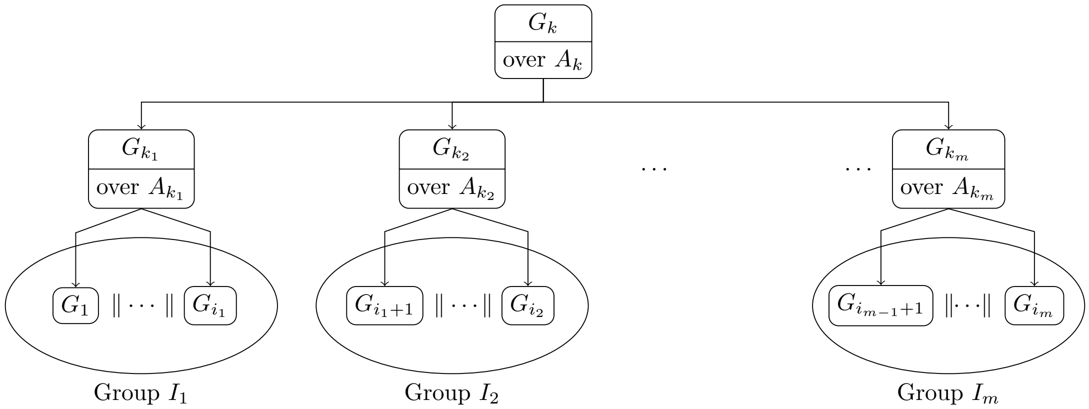

Fig. 3. Multilevel architecture.

Then $\|_{i\in I_j}\sup C^j_{i+k}\|C_{k_j}$ is the resulting nonblocking closed-loop for the group I_r after low-level supervisory control with coordination has been applied.

On the high level we consider the new plants $L_1^{hi} = \sup cC(P_{I_1}(K)), \ldots, L_m^{hi} = \sup cC(P_{I_m}(K))$. The advantage of this choice is that by taking the new (restricted) plants we can actually obtain larger (more permissive) sublanguages that will still remain controllable with respect to the original plant due to the low level computation. For these new plants we then apply standard coordination control of [3]. This means that first of all a high-level coordinator alphabet A_k has to be found so that

$$K = \|_{r=1}^m P_{I_r+k}(K)\,.$$

The corresponding high-level coordinator is computed in the standard way $G_k = P_k(\|_{r=1}^m \sup cC(P_{I_r}(K)))$.

Again, $L(G_k)$ is denoted by L_k and it has to be tested if K is conditionally controllable with respect to generators given by (new high level plant) languages $L_1^{hi} = \sup cC(P_{I_1}(K)), \ldots, L_m^{hi} = \sup cC(P_{I_m}(K))$ computed above and uncontrollable alphabets $A_{I_r,u}$, $i \in I_r$, and $A_{k,u} = A_k \cap A_u$. In the negative case, the supremal conditional sublanguage is computed. The resulting supremal conditionally controllable sublanguage $\sup cC(K,(L_i^{hi})_{i=1,\ldots,m,k},(A_{I_i,u})_{i=1,\ldots,m},A_{k,u})$, denoted $\sup cC(K)$ has the form:

$$\sup cC(K) = \|_{i=1}^m \sup C(P_{I_i+k}(K), L_i^{hi}\|\sup C_k, A_{I_i+k,u}), \quad (3)$$

with $\sup C_k = \sup C(P_k(K), L_k, A_{k,u})$ provided the conditions of Corollary 7 are satisfied.

Similarly as at the low level it may happen that for K that is not prefix-closed, the languages $\sup C(P_{I_i+k}(K), L_i^{hi}\|\sup C_k, A_{I_i+k,u})$, $i = 1,\ldots,m$, computed using Theorem 8 extended to general $n > 1$, denoted by $\sup C_{i+k}$, are conflicting, which causes blocking. Then Theorem 9 can be applied (possibly with A_k extended such that the observer condition is satisfied). A high level coordinator for nonblockingness is then defined by

$$C_k = \|_{i=1}^n P_k(\sup C_{i+k}). \quad (4)$$

Note that in particular

$$\begin{aligned}\sup cC(K,(L_i^{hi})_{i=1,\ldots,m,k},(A_{I_i,u})_{i=1,\ldots,m},A_{k,u}) \\ = \|_{i=1}^m \sup C(P_{I_i+k}(K), L_i^{hi}\|\sup C_k, A_{I_i+k,u}) \quad (5) \\ \subseteq \|_{i=1}^m P_{I_i+k}(K) = K.\end{aligned}$$

Also, the resulting language is still controllable with respect to the original plant. It follows from definition of the involved languages, transitivity of controllability and its preservation under synchronous product for nonconflicting languages and the fact that for any language and projection $L\|P(L) = L$. Hence we have the following result.

Theorem 2: The bottom-up computation scheme described in Procedure 3 below yields a nonblocking solution to the supervisory control problem of imposing global specification $K \subseteq A^*$ in a nonblocking way.

The above description yields the following algorithm that formally describes the bottom-up computation scheme.

Procedure 3:

1) Find low-level coordinator alphabets A_{k_r}, $r = 1,\ldots,m$, containing shared event alphabets $A_{sh,j}$ such that
$$P_{I_r}(K) = \|_{j\in I_r} P_{j+k_r}(K)\,.$$
2) Extend if necessary A_{k_r} such that P_{k_r} are all L_i-observers, $r = 1,\ldots,m$.
3) Compute the low level group coordinators
$$G_{k_r} = P_{k_r}(\|_{\ell=i_{r-1}+1}^{i_r} G_\ell) = \|_{\ell\in I_r} P_{k_r}(G_\ell)$$
and set $L_{k_r} = L(G_{k_r})$.
4) Compute the languages
$$\begin{aligned}&\sup cC(P_{I_r}(K)) \\ &\quad = \sup cC(P_{I_r}(K),(L_i)_{i\in I_r},L_{k_r},(A_{i,u})_{i\in I_r},A_{k_r,u})\end{aligned}$$
using Equation (1) and set high level plants $L_1^{hi} = \sup cC(P_{I_1}(K)), \ldots, L_m^{hi} = \sup cC(P_{I_m}(K))$.
5) If there exists a $r \in \{1,\ldots,m\}$ such that $\sup cC(P_{I_r}(K))$ is blocking, compute the low-level coordinators for nonblocking using Equation (2) and set $L_r^{hi} = L_r^{hi} \parallel C_{k_r}$

6) Find a high-level coordinator alphabet A_k by extending the (high-level) shared alphabet A_{sh} such that $K = \|_{r=1}^{m} P_{I_r+k}(K)$.
7) Compute the high-level coordinator

$$G_k = P_k(\|_{r=1}^{m} L_r^{hi})$$

and set $L_k = L(G_k)$.
8) Compute

$$\sup \mathrm{cC}(K) = \sup \mathrm{cC}(K, (L_i^{hi})_{i=1,\ldots,m,k}, (A_{I_i,u})_{i=1,\ldots,m}, A_{k,u})$$

using Equation (3).
9) If $\sup \mathrm{cC}(K)$ is blocking then compute the high-level coordinator for nonblocking C_k using Equation (4) and set $C_k = A^*$ if $\sup \mathrm{cC}(K)$ is nonblocking
10) Set $\sup \mathrm{cC}(K) \| C_k$ as a solution of multilevel coordination control using the bottom-up approach.

□

It should be noted that the computational complexity of all steps in Procedure 3 is polynomial in relatively small parameters (number of states and events of subsystems combined with coordinators) if the observer conditions that guarantee natural projections being small are all satisfied. Let us recall that the monolithic supervisor synthesis is polynomial in the number of states of a modular system (that is however exponential in the number of components). Here, the subsystems are treated separately (they are only combined with coordinators), which decreases the computational complexity when the number of components is high. Also, conditional decomposability can be checked in a polynomial time in the number of components (unlike decomposability and coobservability) and a polynomial-time algorithm to extend an event set to make a language conditionally decomposable is known ([2]).

Let us recall that for specifications that are not prefix-closed $\sup \mathrm{cC}(K)$ can only be computed if $\sup \mathrm{C}_k \subseteq P_k(\sup \mathrm{C}_{i+k})$ for $i = 1, \ldots, n$, cf. Theorem 8. Otherwise, the optimality with respect to our multilevel scheme is lost.

VI. Example

Let us consider again Example 1. On the low (system) level we divide the four generators into two groups $I_1 = \{1,2\}$ and $I_2 = \{3,4\}$. There will be low-level coordinators G_{k_1} and G_{k_2} coordinating $G_1 \| G_2$ and $G_3 \| G_4$, respectively.

Following the procedure for the bottom-up computation scheme we treat at the bottom level separately $P_{1+2}(K)$ as a specification for $L_{1+2} = L_1 \| L_2$ and $P_{3+4}(K)$ as a specification for $L_{3+4} = L_3 \| L_4$, although it is easy to see that $P_{1+2}(K) \| P_{3+4}(K) \not\subseteq K$ due to e.g. $v_1 a \in P_{1+2}(K) \| P_{3+4}(K) \setminus K$ (but this issue will be fixed at the upper level of hierarchy). Concerning $L_1 \| L_2$, the situation is extremely easy as one can check that $P_{1+2}(K) = L_1 \| L_2$, depicted on Fig. 4. Therefore, no coordination control is needed for I_1 (the first group). For $L_3 \| L_4$, we apply coordination control with $P_{3+4}(K)$ as the specification. $P_{3+4}(K)$ is depicted on Fig. 5. Language $L_3 \| L_4$ is on Fig. 6. It is easy to see that $P_{3+4}(K)$ is not decomposable wrt alphabets A_3 and A_4. Hence, for group I_2 we have to find a coordinator G_{k_2}, and its alphabet A_{k_2}, such that $P_{3+4}(K)$ is conditionally decomposable: $P_{3+4}(K) = P_{3+k_2}(K) \| P_{4+k_2}(K)$. It is sufficient to include event v_1 into A_{k_2} in addition to shared events of G_3 and G_4, i.e., $A_{k_2} = \{c, u, v_1\}$. The corresponding coordinator is then $L_{k_2} = P_{k_2}(L_3 \| L_4) = \{\varepsilon, v_1, v_1 c, u\}$. Since $P_{k_2}(K) = L_{k_2}$ there is no need to compute the supervisor for the coordinator G_{k_2}. We only need to compute supervisors S_3 for $L_3 \| P_{k_2}(K)$ and S_4 for $L_4 \| P_{k_2}(K)$ so that the respective specifications $P_{3+k_2}(K)$ and $P_{3+k_2}(K)$ are met for these plants. It can be checked that $P_{3+4}(K)$ is actually conditionally controllable wrt these plants. Again, we have $P_{3+k_2}(K) = L_3 = L_3 \| P_{k_2}(K)$, hence S_3 is given by $P_{3+k_2}(K)$. Finally, $P_{4+k_2}(K) = \{\overline{v_1 v_2 c}, \overline{v_2 v_1}, \overline{u b_2}\}$. Clearly, $P_{4+k_2}(K) \neq L_4 \| P_{k_2}(K)$, but $P_{4+k_2}(K)$ is controllable wrt $L_4 \| P_{k_2}(K)$, hence S_4 is given by $P_{4+k_2}(K)$ itself (meaning c is disabled after the word $v_2 v_1$: but is not disabled after $v_1 v_2$). We recall that $P_{3+4}(K) = P_{3+k_2}(K) \| P_{4+k_2}(K)$.

Now we proceed to the second step of our scheme and we go up to the high level. Here the new plants are given by $L_1^{hi} = P_{1+2}(K)$ and $L_2^{hi} = P_{3+4}(K)$. As it has already been mentioned, $P_{1+2}(K) \| P_{3+4}(K) \not\subseteq K$, hence the high-level shared alphabet $A_{sh} = (A_1 \cup A_2) \cap (A_3 \cup A_4) = \{c, u\}$ has to be extended to make K conditionally decomposable with respect to $A_1 \cup A_2$, $A_3 \cup A_4$, and A_k. Clearly, it suffices to include a into A_{sh}, i.e., $A_k = \{a, c, u\}$. First of all, the high-level coordinator is given by $L_k = P_k(L_1^{hi} \| L_2^{hi}) = \{\varepsilon, c, a, au\}$. It can be checked that K is conditionally controllable wrt

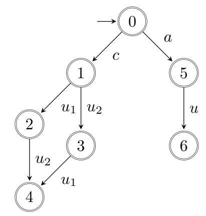

Fig. 4. Generator for $P_{1+2}(K) = L_1 \| L_2$.

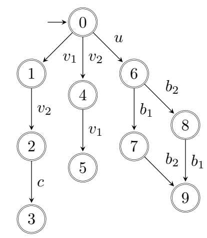

Fig. 5. Generator for $P_{3+4}(K)$.

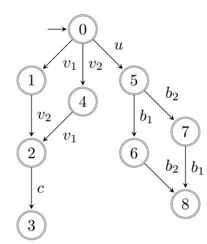

Fig. 6. Generator for $L_3 \| L_4$.

high level plants and the high-level coordinator. We have $L_1^{hi} \| L_k = L_1^{hi}$. It is easy to see that $P_{1+2+k}(K) = P_{1+2}(K) = L_1^{hi}$, i.e., no high-level supervisor S_1^{hi} is needed to impose $P_{1+2+k}(K)$ for the plant $L_1^{hi} \| L_k$. Concerning $L_2^{hi} \| L_k = L \setminus \{v_1 v_2 c u_1, v_1 v_2 c u_1 u_2, v_1 v_2 c u_2, v_1 v_2 c u_2 u_1, v_2 v_1 c\}$ one can check that $P_{3+4+k}(K)$ is controllable with respect to $L_2^{hi} \| L_k$, hence high level supervisor S_2^{hi} is given by $P_{3+4+k}(K)$ itself. ◁

VII. Concluding Remarks

In this paper, coordination control of large modular discrete-event systems has been studied. The coordination control paradigm has been extended to the multilevel coordination with a hierarchical structure of coordinators and supervisors based on a bottom-up approach.

Acknowledgments

The research has been supported by the MŠMT grant LH13012 (MUSIC), and by RVO 67985840.

References

[1] C. G. Cassandras and S. Lafortune, *Introduction to discrete event systems*, 2nd ed. Springer, 2008.

[2] J. Komenda, T. Masopust, and J. H. van Schuppen, “On conditional decomposability,” *Systems Control Lett.*, vol. 61, no. 12, pp. 1260–1268, 2012.

[3] ——, “Supervisory control synthesis of discrete-event systems using a coordination scheme,” *Automatica*, vol. 48, no. 2, pp. 247–254, 2012.

[4] ——, “Multilevel coordination control of modular DES,” in *IEEE CDC*, Florence, Italy, 2013, pp. 6323–6328.

[5] ——, “Coordination control of discrete-event systems revisited,” *Discrete Event Dyn. Syst.*, 2014, to appear.

[6] J. Komenda and J. H. van Schuppen, “Coordination control of discrete event systems,” in *WODES*, Gothenburg, Sweden, 2008, pp. 9–15.

[7] P. J. Ramadge and W. M. Wonham, “Supervisory control of a class of discrete event processes,” *SIAM J. Control Optim.*, vol. 25, no. 1, pp. 206–230, 1987.

[8] K. Schmidt and C. Breindl, “On maximal permissiveness of hierarchical and modular supervisory control approaches for discrete event systems,” in *WODES*, Gothenburg, Sweden, 2008, pp. 462–467.

[9] K. Wong and W. Wonham, “Hierarchical control of discrete-event systems,” *Discrete Event Dyn. Syst.*, vol. 6, no. 3, pp. 241–273, 1996.

[10] W. M. Wonham, “Supervisory control of discrete-event systems,” 2012, lecture notes, University of Toronto, [Online]. Available at http://www.control.utoronto.ca/DES/.

Appendix I
Coordination Control

Basic concepts of coordination control with a single coordinator are recalled. See [5] for further discussion on computational complexity and other issues.

For A_i, A_j, $A_\ell \subseteq A$, we use the notation P_ℓ^{i+j} to denote the projection from $(A_i \cup A_j)^*$ to A_ℓ^*. If $A_i \cup A_j = A$, we simply write P_ℓ. Moreover, $A_{i,u} = A_i \cap A_u$ denotes the set of locally uncontrollable events.

Definition 4 (Conditional decomposability): A language $K \subseteq \cup_{i=1}^n A_i$ is *conditionally decomposable with respect to* $(A_i)_{i=1}^n$ *and* A_k, where $\bigcup_{i,j\in\{1,2,\dots,n\}}^{i\neq j} (A_i \cap A_j) \subseteq A_k \subseteq \bigcup_{j=1}^n A_j$, if

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

for projections P_{i+k} from $\bigcup_{j=1}^n A_j$ to $A_i \cup A_k$. ◁

The problem of coordination control synthesis is now recalled for $n = 2$.

Problem 5: Given generators G_1, G_2 over alphabets A_1, A_2, respectively, and a coordinator G_k over A_k, where $A_1 \cap A_2 \subseteq A_k \subseteq A_1 \cup A_2$. Let $K \subseteq L_m(G_1 \| G_2 \| G_k)$ be a specification such that K and $\overline{K}$ are conditionally decomposable with respect to A_1, A_2, A_k. The problem of coordination control synthesis is to determine nonblocking supervisors S_1, S_2, S_k for the respective generators so that the closed-loop system with the coordinator satisfies

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

◁

It has been shown that conditional controllability together with conditional decomposability form an equivalent condition for a language to be exactly achieved by the closed-loop system within our coordination control architecture.

Definition 6: A language $K \subseteq L(G_1 \| \dots G_n)$ is *conditionally controllable* for generators G_i, $i = 1, dots, n, k$ and uncontrollable alphabets $A_{i,u}$, $i = 1, dots, n, k$ if

1) $P_k(K)$ is controllable wrt $L(G_k)$ and $A_{k,u}$,
2) $P_{i+k}(K)$ is controllable wrt $L(G_i) \parallel \overline{P_k(K)}$ and $A_{i+k,u}$,

where $A_{i+k,u} = (A_i \cup A_k) \cap A_u$, for $i = 1, \dots, n$. ◁

If the specification K fails to be conditionally controllable, then we consider the supremal conditionally controllable sublanguage that always exists, cf. [5].

Below an extension to $n \geq 2$ of [5, Theorem 10] is presented. The reader is invited to see e.g. [8] for definition of observer and LCC properties.

Corollary 7: Let $K \subseteq L = L(G_1 \| \dots \| G_n \| G_k)$ be a prefix-closed language, where G_i is over A_i, $i = 1, \dots, n, k$. Assume that $K = \|_{i=1}^n P_{i+k}(K)$ (K is conditionally decomposable) and define $\sup \mathrm{C}_k = \sup \mathrm{C}(P_k(K), L_k, A_{k,u})$ and $\sup \mathrm{C}_{i+k} = \sup \mathrm{C}(P_{i+k}(K), L_i \| \sup \mathrm{C}_k, A_{i+k,u})$, for $i = 1, \dots, n$. Let P_k^{i+k} be an $(P_i^{i+k})^{-1}(L(G_i))$-observer and LCC for $(P_i^{i+k})^{-1}(L(G_i))$, for $i = 1, \dots, n$. Then,

$$\sup \mathrm{cC}(K, L, (A_{1,u}, \dots, A_{n,u}, A_{k,u})) = \|_{i=1}^n \sup \mathrm{C}_{i+k}. \quad (6)$$

■

Another result on how to compute $\sup \mathrm{cC}$ is extended to $n \geq 2$.

Theorem 8: [5, Theorem 6] Consider the setting of Problem 5, and the languages $\sup \mathrm{C}_{i+k}$, $i = 1, \dots, n$ and $\sup \mathrm{C}_k$ defined in Corollary 7, where K is possibly not prefix-closed. If for all i: $\sup \mathrm{C}_k \subseteq P_k(\sup \mathrm{C}_{i+k})$, then $\sup \mathrm{cC}(K, L, (A_{1,u}, \dots, A_{n,u}, A_{k,u})) = \|_{i=1}^n \sup \mathrm{C}_{i+k}$. ■

Similarly, we extend [5, Theorem 7] to general $n \geq 2$.

Theorem 9: Consider a modular plant with local marked languages $L_i = L_m(G_i) \subseteq A_i^*$, $i = 1, \dots, n$, and let projection $P_k : A^* \to A_k^*$, with shared events included in A_k, be an L_i-observer, for $i = 1, \dots, n$. Define C_k as the nonblocking generator with $L_m(C_k) = \|_{i=1}^n P_k(L_i)$ with notation $L_k = L_m(C_k)$, i.e., $L(C_k) = \overline{L_k} = \overline{\|_{i=1}^n P_k(L_i)}$. Then the coordinated system $G \| C_k$ is nonblocking, i.e., $\overline{\|_{i=1}^n L_i \| L_m(C_k)} = \|_{i=1}^n \overline{L_i} \| \overline{L_m(C_k)}$.