

# Optimal Solutions of Modular Supervisory Control Problems with Indecomposable Specification Languages

Jan Komenda

*Institute of Mathematics, Czech Academy of Sciences,  
Brno Branch, Zizkova 22, 616 62 Brno, Czech Republic*

`komenda@ipm.cz`

Jan H. van Schuppen

*CWI, P.O. Box 94079,  
1090 GB Amsterdam, The Netherlands*

`J.H.van.Schuppen@cwi.nl`

**Abstract**—This paper concerns supervisory control of large-scale modular discrete event systems (DES) with partial observations and general (indecomposable) specification languages. Recently we have found new methods for computing supremal controllable sublanguages and supremal normal sublanguages independently. Unfortunately, these methods are difficult to put simply together for computation of supremal controllable and normal sublanguages. Therefore, we propose first a new method for computation of supremal controllable sublanguages that can be viewed as a complete observation counterpart of a method we have developed for supremal normal sublanguages. These can be put together in a procedure for computing optimal sublanguages, which avoids building of the global plant. Unlike our previous results, we present both necessary and sufficient structural conditions for modular control synthesis to equal global control synthesis for partially observed modules and general (indecomposable) specification languages.

**Keywords**—Modular DES, indecomposable specifications, supremal controllable and normal sublanguages

## I. INTRODUCTION

Large DES are typically composed of concurrent subsystems, also called modules. Modular control of DES represented by finite automata ([9]) has been introduced by P.J. Ramadge and W.M. Wonham in [12]. The overall systems are formed as a parallel compositions (i.e. synchronous product) of local components. There exists an extensive literature on this important topic. In earlier work the input alphabets of the local components were identical ([12]). In [10] and [2] quite a restrictive condition is imposed on events shared by several local components: shared events must be controllable for all subsystems. This assumption has been generalized in [11] to the condition that the shared events must have the same control status for all subsystems that share a particular event. Specification languages for the global plant that are not decomposable into local specification languages are considered in this paper.

In the case of decomposable specifications and complete observations, conditions for local control synthesis without loss of optimality are established in [11]. A sufficient condition, called mutual controllability, is used to cover the case, where not all shared events are controllable, but their controllability status is constant. We have extended this result to the case of modules with partial observations in [7], where a structural condition of mutual normality is proposed for distributivity of supremal normal sublanguages with the

synchronous product. In [3] these results are put together and the sufficient conditions are obtained for distributivity between supremal controllable and normal sublanguages and the synchronous product of languages. The proof is based on a single-step algorithm for supremal controllable and normal sublanguages developed in [6].

For the case of global (indecomposable) specifications an approach has been proposed in [2]. It is based on partial controllability and works only when all shared events are controllable. This result has been extended in [4] to the case, where the controllability status of a shared event is constant (i.e. does not depend on the subsystems that share a particular event). However, it is by no means obvious how to obtain a partial observations counterpart of partial controllability. There is of course a natural candidate property to be called partial normality that has similar properties as partial controllability, but it fails to be useful for computation of supremal normal sublanguages. Therefore we have chosen a second option: finding a complete observation counterpart of the method developed in [5] for computation of supremal normal sublanguages. We introduce global mutual controllability, which is a sufficient condition, but also modular controllability, which is necessary among all structural conditions for distributivity between supremal controllable sublanguages and language intersections. Based on the new algorithm for the computation of supremal controllable sublanguages and the single-step algorithm from [4] for the modular computation of supremal normal sublanguages we propose a new algorithm for modular computation of supremal controllable and normal sublanguages. The advantage of the method we propose is twofold: (1) Avoided is the construction of the global plant. Note that the only objects used are the local plants and the global specification. (2) Avoided is the iterative computation of supremal controllable sublanguages and supremal normal languages. A direct computation of these sublanguages is proposed.

This paper is organized as follows. In the next section the framework of modular control with partial observations is recalled. In section 3 a novel method is presented for computation of supremal controllable sublanguages. In section 4 the single step algorithm for computation of supremal controllable and normal sublanguages is recalled from [6]. The method developed in section 3 is used in section 5, where a formula and underlying method for computation of supremal controllable and normal sublanguages are pro-

posed. Necessary definitions and theorems on coalgebra and coinduction can be found in the appendix.

## II. MODULAR CONTROL WITH PARTIAL OBSERVATIONS AND GLOBAL SPECIFICATIONS

Maximally permissive modular supervision of the concurrent behavior of local subplants (partial automata)  $G_1, \dots, G_n$  is studied. The notation  $\mathbb{Z}_n = \{1, 2, \dots, n\}$  is used. The coalgebraic framework of [8] and [6] is used, i.e.  $G_i, i \in \mathbb{Z}_n$  are partial automata and their behaviors are partial languages. Recall that a partial automaton is nothing but a generator of a DES with a partial transition function and that a partial language is a tuple of a marked language and a closed language. The partial language associated with  $G_i, i \in \mathbb{Z}_n$  are denoted by  $L_i = (L_i^1, L_i^2)$  (see appendix). We assume that the global DES is the synchronous product  $G = \parallel_{i=1}^n G_i$  of local subsystems. The behavior of  $G$  is the partial language denoted by  $L$ . Denote  $A = \cup_{i=1}^n A_i$  the global alphabet and  $P_i : A^* \rightarrow A_i^*$  the projections to the local alphabets. The concept of inverse projection:  $P_i^{-1} : \text{Pwr}(A_i^*) \rightarrow \text{Pwr}(A^*)$  is also used.

We denote the global plant and the specification (partial) languages by  $L = L(G)$  and  $K$ , respectively. In our modular setting,  $L$  is decomposable into local plant languages:  $L = L_1 \parallel \dots \parallel L_n$ .  $K$  is usually similarly decomposable into local specification languages  $K_i \subseteq L_i$ . The general case is when  $K$  is given only as a global specification and moreover  $K$  may not be included in  $L$ . This has been considered in [2] for complete observations under restrictive structural conditions (all shared events are controllable). Following [2], we consider for any  $i \in \mathbb{Z}_n$  languages  $K_i := K \cap P_i^{-1}(L_i)$ , which will play the role of local specification in the case of indecomposable specification  $K$ . Note however that  $K_i$  is a partial language over the global alphabet  $A$ , in particular it is different from  $P_i(K)$  that appear if the specification is decomposable, (cf. [7]).

As customary in modular supervisory control, local alphabets of the local subplants, denoted by  $A_i, i \in \mathbb{Z}_n$  and not necessarily pairwise disjoint, are such that  $A_i = A_{iu} \cup A_{ic}$ , where  $A_{iu}$  stands for the subset of locally uncontrollable events and  $A_{ic}$  stands for the subset of locally controllable events. We assume that

$$\forall i \neq j, A_{iu} \cap A_j = A_i \cap A_{ju}. \quad (1)$$

This assumption originally stemming from [11] means that the events shared by two local subsystems must have the same control status for both controllers associated to these subsystems. We denote by  $A_c = \cup_{i=1}^n A_{ic}$  and  $A_u = A \setminus A_c$  the subsets of globally controllable and globally uncontrollable events, respectively. We still have  $A_u = \cup_{i=1}^n A_{iu}$  due to the assumption (1).

Recall that for a specification language to be exactly achieved by a supervisor, a property called controllability is required. A (partial) language  $K$  is said to be controllable with respect to  $L$  and  $A_u$ , if

$$K^2 A_u \cap L^2 \subseteq K^2.$$

For a specification that is not controllable, optimal approximations are considered. Since the specification mostly represents some safety constraints, controllable sublanguages have been studied. Supremal controllable sublanguage always exists, because controllability is preserved by unions.

Let  $A_i = A_{o,i} \cup A_{uo,i}$  be the decomposition of local events into locally observable ( $A_{o,i}$ ) and locally unobservable ( $A_{uo,i}$ ) event subsets. Note a difference in notation between controllability and observability status of local events. This notation is chosen to avoid confusion between locally uncontrollable event set  $A_{iu}$  and locally unobservable event set  $A_{uo,i}$ . The global system has the observation set  $A_o = \cup_{i=1}^n A_{o,i} \subseteq A = \cup_{i=1}^n A_i$ . Globally unobservable events are denoted by  $A_{uo} = A \setminus A_o$ . We assume similarly  $\forall i \neq j \in \{1, \dots, n\}$  we have  $A_{o,i} \cap A_j = A_i \cap A_{o,j}$ , i.e. observational status of a shared event must be the same for all modules that share a particular event. Therefore we have also  $A_{uo} = \cup_{i=1}^n A_{uo,i}$ . The corresponding global projection that erases unobservable events is denoted by  $P : A^* \rightarrow A_o^*$ .

Partial observations in individual modules are expressed via local projections from  $A_i^*$  to  $A_{o,i}^*$ , but due to the handling of general indecomposable (global) specification languages only the afore mentioned global projection  $P : A^* \rightarrow A_o^*$  is considered in this paper.

An additional property of observability is required for a specification to be exactly achieved when partial observations are considered. In this paper we are interested in a slightly stronger property of normality, which is equivalent to observability if  $A_c \subseteq A_o$ , but is closed under unions. Recall that  $K$  is called  $(L, P)$ -normal if  $K^2 = P^{-1}P(K^2) \cap L^2$ .

Therefore supremal normal, and supremal controllable and normal sublanguages always exist. These are often considered as optimal solutions of supervisory control problems with partial observations and they are indeed optimal in the case  $A_c \subseteq A_o$ , a much weaker assumption than  $\forall i \in \mathbb{Z}_n : A_{ic} \subseteq A_{o,i}$ .

## III. SUPREMAL CONTROLLABLE SUBLANGUAGES

Before dealing with the main problem of this paper we establish a novel formula and associated method for computing supremal controllable sublanguage of a global specification language. The coinductive proof of the formula for supremal controllable sublanguage is based on coinductive definition of supremal controllable sublanguage [6]. The coinductive definition of  $\text{supC}(K \cap L, L, A_u)$  can be found in the appendix.

The following concept that we call global mutual controllability will be used.

*Definition 3.1 (Global mutual controllability):* Local plant (partial) languages  $L_i, i \in \mathbb{Z}_n$  are called *globally mutually controllable* if for all  $i, j \in \mathbb{Z}_n$  with  $i \neq j$  we have

$$P_j^{-1}(L_j^2)A_{ju} \cap P_i^{-1}(L_i^2) \subseteq P_j^{-1}(L_j^2).$$

Note that this concept is stronger than that of [4], where local uncontrollable events  $A_{ju}$  are replaced by local shared uncontrollable events  $A_{ju} \cap A_i$ .

A second concept that we call modular controllability is needed.

*Definition 3.2 (modular controllability):* Local plant (partial) languages  $L_i$ ,  $i \in \mathbb{Z}_n$  are called *modularly controllable* if for any  $i \in \mathbb{Z}_n$  we have

$$(L^2 A_u) \cap P_i^{-1}(L_i^2) \subseteq L^2.$$

This condition is in general weaker than global mutual controllability and it will play the role of a necessary condition. Note that modular controllability is not comparable to what we have called in [4] global mutual controllability, but is much weaker than that of strong global mutual controllability used in [4]. In global mutual controllability, local uncontrollable events  $A_{iu}$  are replaced by local shared uncontrollable events  $A_{ju} \cap A_i$ , but also  $L$  is replaced by  $P_j^{-1}(L_j)$ .

Our main theorem follows. It provides *sufficient and necessary* structural conditions for modular control synthesis to equal global control synthesis in the case of complete observations and of indecomposable specifications.

*Theorem 3.1:* Suppose that  $\forall i, j \in \mathbb{Z}_n$  with  $i \neq j$  we have  $A_{iu} \cap A_j = A_i \cap A_{ju}$ .

- (a) If  $L_i$ ,  $i \in \mathbb{Z}_n$  are globally mutually controllable then

$$\begin{aligned} \sup C(K \cap L, L, A_u) = & \quad (2) \\ \bigcap_{i=1}^n \sup C(K_i, P_i^{-1}(L_i), A_u). \end{aligned}$$

- (b) If for a given modular plant equation (2) holds for any global specification  $K$  then local plant (partial) languages  $L_i$ ,  $i \in \mathbb{Z}_n$  are modularly controllable.

*Remark 3.2:* Note that the second part of the statement in the previous theorem means that modular controllability is necessary structural condition among all structural conditions, because equation (2) is required to hold for any specification language  $K$ .

*Proof:* (a) The coinductive proof principle will be used for sufficiency, i.e. it is sufficient to show that under the conditions listed above the following relation:

$$\begin{aligned} R = \{ \langle (K \cap L) /_{A_u}^{SC} L, \\ \bigcap_{i=1}^n [K_i /_{A_u}^{SC} P_i^{-1}(L_i)] \rangle; K, L \in \mathcal{L} \} \end{aligned}$$

is a bisimulation relation, from which the equality follows by coinduction (see Appendix).

(b) The necessity of modular controllability is easy to show: Let  $L_i$ ,  $i \in \mathbb{Z}_n$  be given and equation (2) holds for any  $K \subseteq A^*$ . Then in particular (the more difficult) inclusion

$$\begin{aligned} \sup C(K \cap L, L, A_u) \subseteq \\ \bigcap_{i=1}^n \sup C(K_i, P_i^{-1}(L_i), A_u) \end{aligned}$$

holds for any  $K \subseteq A^*$ . This is equivalent to  $\sup C(K \cap L, L, A_u) \subseteq \sup C(K_i, P_i^{-1}(L_i), A_u)$  for any  $K \subseteq A^*$  and  $i \in \mathbb{Z}_n$ . The inclusion holds in particular for  $K = L$ , which entails  $K_i = L$  for any  $i \in \mathbb{Z}_n$ . Hence,

$L = \sup C(K \cap L, L, A_u) \subseteq \sup C(L, P_i^{-1}(L_i), A_u)$ . Since  $\sup C(L, P_i^{-1}(L_i), A_u) \subseteq L$  by definition of the  $\sup C$  operator, we obtain  $L = \sup C(L, P_i^{-1}(L_i), A_u)$ . But this means that  $L$  is controllable with respect to  $P_i^{-1}(L_i)$  and  $A_u$  for any  $i \in \mathbb{Z}_n$ , which is the modular controllability condition. ■

Finally, we discuss different properties used in this section. First of all note that global mutual controllability implies modular controllability, which should obviously be true in view of the last theorem. In order to see this directly, we first establish the following property.

*Proposition 3.3:* Global mutual controllability (GMC) is equivalent to the following property: for any  $i \neq j \in \mathbb{Z}_n$  we have

$$P_j^{-1}(L_j^2) A_u \cap P_i^{-1}(L_i^2) \subseteq P_j^{-1}(L_j^2).$$

*Proof:* Note that the only difference is that  $A_{ju}$  in GMC is replaced by  $A_u$ . Therefore the new property is clearly stronger than GMC. Surprisingly the converse implication is satisfied as well. Suppose that GMC holds,  $s \in P_j^{-1}(L_j^2)$ ,  $u \in A_u$ , and  $su \in P_i^{-1}(L_i^2)$ . Then we have two cases: either  $u \in A_j$  or  $u \notin A_j$ . The former case entails that  $u \in A_{ju}$  due to shared event controllability status assumption. Using GMC we conclude  $su \in P_j^{-1}(L_j^2)$ . In the latter case we notice that  $P_j(u) = \varepsilon$ , i.e.  $P_j(su) = P_j(s)$ , which means that  $P_j(su) \in L_j^2$ , i.e.  $s \in P_j^{-1}(L_j^2)$ . ■

Now, let for any  $i, j \in \mathbb{Z}_n$  with  $i \neq j$ :

$$P_j^{-1}(L_j^2) A_u \cap P_i^{-1}(L_i^2) \subseteq P_j^{-1}(L_j^2).$$

Since for  $j = i$  the inclusion is trivially true, we have the inclusion for any  $i, j \in \mathbb{Z}_n$ . Then, by taking intersection for  $j$  over  $\mathbb{Z}_n$  we obtain:

$$\bigcap_{j=1}^n [P_j^{-1}(L_j^2) A_u \cap P_i^{-1}(L_i^2)] \subseteq \bigcap_{j=1}^n P_j^{-1}(L_j^2).$$

Since  $\bigcap_{i=1}^n [P_j^{-1}(L_j^2)] A_u \subseteq \bigcap_{i=1}^n [P_j^{-1}(L_j^2) A_u]$  we obtain finally,  $L^2 A_u \cap P_i^{-1}(L_i^2) \subseteq L^2$ , i.e. modular controllability (MC). Interestingly, if in MC we replace  $A_u$  by  $A_{iu}$  we obtain a strictly weaker condition than MC, unlike GMC. Another interesting observations is that for  $n = 2$ , then GMC is equivalent to MC. Indeed, let MC holds. We show that

$$P_j^{-1}(L_j^2) A_u \cap P_i^{-1}(L_i^2) \subseteq P_j^{-1}(L_j^2).$$

Let  $s \in P_j^{-1}(L_j^2)$ ,  $u \in A_u$ , and  $su \in P_i^{-1}(L_i^2)$ . Then we have  $s \in P_i^{-1}(L_i^2)$  as well, because  $P_i^{-1}(L_i^2)$  is prefix-closed. Since we have two modules and  $i \neq j$  we obtain  $s \in L^2$ . Now using MC

$$su \in L^2 A_u \cap P_i^{-1}(L_i^2) \subseteq L^2 \subseteq P_j^{-1}(L_j^2),$$

whence  $su \in P_j^{-1}(L_j^2)$  and GMC holds. For a number of modules greater than or equal to 3 however, GMC and MC differ, because we cannot deduce  $s \in L^2$  from  $s \in P_j^{-1}(L_j^2)$ ,  $u \in A_u$ , and  $su \in P_i^{-1}(L_i^2)$ .

#### IV. SUPREIMAL CONTROLLABLE AND NORMAL SUBLANGUAGES

In this section we recall a single step algorithm for computation of supremal controllable and normal sublanguages from [5]. The concept of  $\varepsilon$ -transitions is needed to formulate an observational indistinguishability relation due to partial observations. Generators of DES are partial automata  $S = (S, \langle o, t \rangle)$  (see appendix for more details).

*Definition 4.1:* ( $\varepsilon$ -transition.) For  $s \in S$  we define  $s \xrightarrow{\varepsilon} s'$  if  $\exists \tau \in A_{uo}^* : s \xrightarrow{\tau} s_\tau = s'$ .

Denote the initial state of the DES generator  $S$  by  $s_0$ . An auxiliary concept that reflects the fact that due to partial observations it is not possible to distinguish between states is recalled from [6] :

*Definition 4.2:* (Observational indistinguishability relation on  $S$ .) A binary relation  $Aux(S)$  on  $S$ , called the *observational indistinguishability relation* is the smallest relation satisfying:

- (i)  $\langle s_0, s_0 \rangle \in Aux(S)$
- (ii)  $\forall \langle s, t \rangle \in Aux(S) : (s \xrightarrow{\varepsilon} s' \text{ and } t \xrightarrow{\varepsilon} t') \Rightarrow \langle s', t' \rangle \in Aux(S)$
- (iii)  $\forall \langle s, t \rangle \in Aux(S) \text{ and } \forall a \in A_o : (s \xrightarrow{a} s_a \text{ and } t \xrightarrow{a} t_a) \Rightarrow \langle s_a, t_a \rangle \in Aux(S)$ .

Next we recall the concept of state-partition automaton, which we define here as follows:

*Definition 4.3:* (State-partition automaton) A partial automaton  $S$  is called a state-partition automaton if  $\forall s \in S : s = (s_0)_{w_1} = (s_0)_{v_1}$  for  $w_1 \in A^*$  and  $v_1 \in A^*$  and  $\forall s' \in S : s' = (s_0)_{w_2}$  for  $w_2 \in A^*$  with  $P(w_2) = P(v_1)$  there exists  $w_2 \in A^*$  with  $P(w_2) = P(w_1)$  and  $s' = (s_0)_{w_2}$ .

We assume that a partial language  $L$  is represented by a partial automaton  $S$  with initial state  $s_0$ . We mean by this that the corresponding behavior homomorphisms  $l : S \rightarrow \mathcal{L}$  (see appendix) satisfies  $l(s_0) = L$ . A characterization of  $Aux(S)$  follows:

*Lemma 4.1:* For any  $s, s' \in S : \langle s, s' \rangle \in Aux(S)$  if and only if there exist  $w, w' \in L^2$  such that  $P(w) = P(w')$ ,  $s = (s_0)_w$  and  $s' = (s_0)_{w'}$ . Moreover, if  $S$  is a state-partition automaton then  $\forall v \in L^2$  and  $s' \in S$  we have  $\langle (s_0)_v, s' \rangle \in Aux(S)$  if and only if there exists  $v' \in L^2$  such that  $P(v) = P(v')$  and  $s' = (s_0)_{v'}$ .

Local plant languages will be denoted by  $L_i, i \in \mathbb{Z}_n$ . We assume that global specification  $K$  is not decomposable into local specifications. Recall from [1] that  $K$  is called  $(L, P)$ -normal if  $K^2 = P^{-1}P(K^2) \cap L^2$ . It is known that normal languages are closed under unions, hence supremal

normal sublanguages always exist.

*Remark 4.2:* An order relation on partial languages induced by second components only is used: we write  $K \subseteq L$  iff  $K^2 \subseteq L^2$ .

We denote the supremal sublanguage of  $K$  which is both  $(L, A_u)$ -controllable and  $(L, P)$ -normal sublanguage by  $\sup \text{CN}(K, L, A_u, A_o)$ . Since  $K$  is not in general included in  $L$  we investigate  $\sup \text{CN}(K \cap L, L, A_u, A_o)$ .

We are looking for conditions that enable computation of  $\sup \text{CN}(K \cap L, L, A_u, A_o)$  without having to build the recognizer of  $L$  itself, i.e. using only given  $K$  and  $P_i^{-1}(L_i), i \in \mathbb{Z}_n$ . In view of our previous results [4] and [5] the best way is to find a formula for  $\sup \text{CN}(K \cap L, L, A_u, A_o)$  as the intersection of  $\sup \text{CN}$  operators that do not involve  $L$ . In order to prove such a formula, a single step algorithm for computing supremal controllable and normal sublanguages turns out to be very useful. We recall it from [6]:

*Algorithm 1:* Let automata  $S$  and  $T$  representing  $K \cap L$  and  $L$ , respectively be such that  $S$  is a subautomaton of  $T$  and  $S$  is a state-partition automaton. The transition functions of  $S$  and  $T$  are denoted by  $\rightarrow$  and  $\rightarrow_1$ , respectively. Let us construct the partial automaton  $\tilde{S} = (\tilde{S}, \langle \tilde{o}, \tilde{t} \rangle)$  with  $\tilde{t}$  denoted by  $\rightarrow_1$ .

Define the auxiliary condition (\*) as follows:  
if  $a \in A_u \cup A_{uo}$  then  $\forall u \in (A_u \cup A_{uo})^* : s_a \xrightarrow{u} \Rightarrow s_a \xrightarrow{u}_1$ ;  
if  $a \in A_c \cap A_o$  then  $\forall s' \approx_{Aux(S)} s : s' \xrightarrow{a} \Rightarrow s' \xrightarrow{a}_1$ , in which case also  $\forall u \in (A_u \cup A_{uo})^* : s'_a \xrightarrow{u} \Rightarrow s'_a \xrightarrow{u}_1$ .

Below are the steps of the algorithm.

1. Let  $\tilde{S} := \{s_0\}$ .
2. For any  $s \in \tilde{S}$  and  $a \in A$  we put  $s \xrightarrow{a}, s_a$  if  $s \xrightarrow{a}_1$  and condition (\*) is satisfied; and we put in the case  $s \xrightarrow{a}$ , also  $\tilde{S} := \tilde{S} \cup \{s_a\}$ .
3. For any  $s \in \tilde{S}$  we put  $\tilde{o}(s) = o(s)$ .

As usual, we denote by  $\tilde{l}$  the unique (behavior) homomorphism given by finality of  $\mathcal{L}$ . We can verify by coinduction that [6]:

*Theorem 4.3:*  $\tilde{l}(s_0)$  is the supremal controllable (with respect to  $L$  and  $A_u$ ) and  $(L, P)$ -normal sublanguage of  $K \cap L$ .

#### V. MODULAR COMPUTATION OF SUPREIMAL CONTROLLABLE AND NORMAL SUBLANGUAGES

In this section we propose a formula and an associated algorithm for modular computation of supremal controllable and normal sublanguages based on the formula of Theorem 3.1 and a similar formula in [5]. Let us first recall from [5] a structural condition called global mutual normality. It is similar to the mutual normality in the case of decomposable specification ([7]), but concerns  $P_i^{-1}(L_i)$  instead of  $L_i$  themselves.

*Definition 5.1 (Global mutual normality):* The local plant (partial) languages  $L_i \subseteq (A_i^* \times A_i^*)$ ,  $i \in \mathbb{Z}_n$  are called *globally mutually normal* if for any  $i, j \in \mathbb{Z}_n$ ,  $i \neq j$  we have

$$(P^{-1}PP_j^{-1})(L_j^2) \cap P_i^{-1}L_i^2 \subseteq P_j^{-1}L_j^2.$$

Another condition is needed that will play the role of necessary a condition.

*Definition 5.2 (modular normality):* Local plant (partial) languages  $L_i$ ,  $i \in \mathbb{Z}_n$  are called *modularly normal* if for any  $i \in \mathbb{Z}_n$  we have that  $L$  is  $(P_i^{-1}(L_i), P)$ -normal, i.e.  $P^{-1}P(L^2) \cap P_i^{-1}(L_i) \subseteq L^2$ .

This condition is in general weaker than global mutual normality (GMN). It is easy to show that

*Proposition 5.1:* Global mutual normality (GMN) implies modular normality (MN).

*Proof:* Let GMN holds, i.e.  $i, j \in \mathbb{Z}_n$ ,  $i \neq j$  we have

$$(P^{-1}PP_j^{-1})(L_j^2) \cap P_i^{-1}L_i^2 \subseteq P_j^{-1}L_j^2.$$

Since for  $i = j$  the inclusion becomes trivial, we may assume that the inclusion is satisfied for any  $i, j \in \mathbb{Z}_n$ . By intersecting both sides for  $j$  ranging in  $\mathbb{Z}_n$  and using the properties of projection and inverse projection we obtain:

$$\begin{aligned} P^{-1}PL^2 &\subseteq \bigcap_{j=1}^n (P^{-1}PP_j^{-1})(L_j^2) \\ &\cap P_i^{-1}L_i^2 \subseteq \bigcap_{j=1}^n P_j^{-1}L_j^2 = L^2, \end{aligned}$$

i.e. MN holds.  $\blacksquare$

Our main result can now be formulated.

*Theorem 5.2:* (Sufficient structural conditions for modular synthesis with global specification to equal global synthesis) Let  $i, j \in \mathbb{Z}_n$ ,  $i \neq j$  we have  $A_{o,i} \cap A_j = A_i \cap A_{o,j}$  and  $A_{c,i} \cap A_j = A_i \cap A_{c,j}$ .

(a) If  $L_i$ ,  $i \in \mathbb{Z}_n$  are globally mutually controllable and globally mutually normal then

$$\begin{aligned} \sup \text{CN}(K \cap L, L, A_u, A_o) &= \quad (3) \\ &= \bigcap_{i=1}^n \sup \text{CN}(K_i, P_i^{-1}(L_i), A_u, A_o). \end{aligned}$$

(b) If for a given modular plant equation (3) holds for any global specification  $K$  then local plant (partial) languages  $L_i$ ,  $i \in \mathbb{Z}_n$  are modularly controllable and modularly normal.

*Proof:* Algorithm 1 is used for the computation of  $\sup \text{CN}(K \cap L, L, A_u, A_o)$  and also for computation of  $\sup \text{CN}(K_i, P_i^{-1}(L_i), A_u, A_o)$  with the first 2 parameters represented by partial automata  $S_i$  and  $T_i$ . Let  $S$  representing  $K \cap L$  and  $T$  representing  $L$  be the same as in Algorithm 1. Algorithm 1 yields the partial automaton  $\tilde{S} = (\tilde{S}, \langle \tilde{o}, \tilde{t} \rangle)$ , subautomaton of  $S$ , that represents  $\sup \text{CN}(K \cap L, L, A_u, A_o)$  according to Theorem 4.3.

Similarly, let partial automata  $S_i = (S_i, \langle o_{1i}, t_{1i} \rangle)$  and  $T_i = (T_i, \langle o_i, t_i \rangle)$  representing  $K_i$  and  $P_i^{-1}(L_i)$ ,  $i \in \mathbb{Z}_n$ ,

respectively, be such that for all  $i \in \mathbb{Z}_n$ :  $S_i$  is a subautomaton of  $T_i$ , and  $S_i$  is a state-partition automaton. The common initial state of these automata is denoted by  $s_0^i$  and the transition functions  $t_{1i}$  and  $t_i$  are denoted by  $\rightarrow_{1i}$  and  $\rightarrow_i$ , respectively. Denote by  $Aux(S_i)$  the observation indistinguishability relation with respect to the projection  $P$ . Algorithm 1 is used to construct partial automata  $\tilde{S}_i = (\tilde{S}_i, \langle \tilde{o}_i, \tilde{t}_i \rangle)$ , subautomata of  $S_i$ , with  $\tilde{t}_i$  denoted by  $\rightarrow_{i}$ . According to Theorem 4.3,  $\tilde{S}_i$  represents the "local"  $\sup \text{CN}(K_i, P_i^{-1}(L_i), A_u, A_o)$ . This enables us to consider the behaviors of the corresponding output automata  $\tilde{S}$  and  $\tilde{S}_i$  of Algorithm 1 with the corresponding parameters detailed above. The coinductive proof principle is used (see appendix). We show that

$$R = \{ \{ [\tilde{l}(s_0)]_w, [\bigcap_{i=1}^n \tilde{l}_i(s_0^i)]_w \} \mid w \in (\tilde{l}(s_0))^2 \}$$

is a bisimulation relation, from which the claim of the theorem follows by coinduction. The necessity of modular controllability and modular normality is easy to show along the same lines as in the complete observation case.  $\blacksquare$

*Remark 5.3:* It is easy to see that (iii) of the above proof (corresponding to the inclusion  $\bigcap_{i=1}^n \sup \text{CN}(K_i, P_i^{-1}(L_i), P_i^{loc}, P) \subseteq \sup \text{CN}(K \cap L, L, A_u, A_o)$ ) has been proven without any assumption.

The last theorem is very useful, because we have found structural conditions under which supremal controllable and normal sublanguages can be computed without manipulating with the global plant  $L$ , but using only  $P_i^{-1}(L_i)$  for  $i \in \mathbb{Z}_n$ . Notice that the assumption  $A_c \subseteq A_o$  is not needed in the theorem. However, if this condition is not satisfied, then the supremal controllable and normal sublanguages are not necessarily optimal solutions. In such a situation closed-loop languages using antipermissive control policy can be considered. Similar conditions can be formulated, in terms of global mutual observability and modular observability.

## VI. CONCLUSION

A new method for modular computation of supremal controllable and normal sublanguages of indecomposable specification languages has been presented. The method is based on a new formula for supremal controllable sublanguage and our recent method for computation of supremal normal sublanguages.

The structural conditions of our methods do not depend on the particular specification, which is very important because general indecomposable specifications are studied. Further research is needed in order to improve our results: the sufficient conditions we have obtained might be weakened or different methods are to be proposed.

## REFERENCES

- [1] S.G. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems*, Kluwer Academic Publishers, 1999.
- [2] B. Gaudin and H. Marchand. Modular Supervisory Control of a Class of Concurrent Discrete Event Systems. Proceedings *WODES'04*, Workshop on Discrete-Event Systems, pp. 181-186, Reims, September 22-24, 2004.

- [3] J. Komenda. Modular Control of Large Distributed Discrete-Event Systems with Partial Observations. In Proceedings of the 15th International Conference on Systems Science, Vol. II, pp. 175-184, Wroclaw, Poland, September 9, 2004..
- [4] J. Komenda, J.H. van Schuppen, B. Gaudin, H. Marchand. Modular Supervisory Control with General Indecomposable Specification Languages. Proceedings of Joint 44th IEEE Conference on Decision and Control and European Control Conference, Sevilla, 12-15.12, 2005.
- [5] J. Komenda and J.H. van Schuppen. Supremal Sublanguages of General Specification Languages Arising in Modular Control of Discrete-Event Systems. Proceedings of Joint 44th IEEE Conference on Decision and Control and European Control Conference, Sevilla, 12-15.12, 2005.
- [6] J. Komenda and J. H. van Schuppen: Control of Discrete-Event Systems with Partial Observations Using Coalgebra and Coinduction. Discrete Event Dynamical Systems: Theory and Applications 15(3), 257-315, 2005.
- [7] J. Komenda and J.H. van Schuppen. Supremal Normal Sublanguages of Large Distributed Discrete-Event Systems. Proceedings *WODES'04*, pp. 73-78, Reims, September 22-24, 2004.
- [8] J.J.M.M. Rutten. Coalgebra, Concurrency, and Control. *Research Report CWI, SEN-R9921*, Amsterdam, November 1999. Available also at <http://www.cwi.nl/~janr>.
- [9] P.J. Ramadge and W.M. Wonham. The Control of Discrete-Event Systems. *Proc. IEEE*, 77:81-98, 1989.
- [10] Y. Willner and M. Heymann. Supervisory Control of Concurrent Discrete-Event Systems. *International Journal of Control*, 54:1143-1166, 1991.
- [11] K.C. Wong and S. Lee. Structural Decentralized Control of Concurrent Discrete-Event Systems. *European Journal of Control*, 8:477-491, 2002.
- [12] W.M. Wonham and P.J. Ramadge. Modular Supervisory Control of Discrete-Event Processes, *Mathematics of Control, Signal and Systems*, 1:13-30, 1988.

## APPENDIX

Coalgebras are categorical duals of algebras (the corresponding functor operates from a given set rather than to a given set). The concept of final coalgebras enables definitions and proofs by coinduction.

### A. Partial automata

In this section partial automata as generators of DES are formulated coalgebraically as in [8]. Final coalgebra of partial automata, i.e. a partial automaton of partial languages is then recalled. Let  $A$  be the set of events. The empty string will be denoted by  $\varepsilon$ . Denote by  $1 = \{\emptyset\}$  the one element set and by  $2 = \{0, 1\}$  the set of Booleans. A partial automaton is a pair  $S = (S, \langle o, t \rangle)$ , where  $S$  is a set of states, and a pair of functions  $\langle o, t \rangle : S \rightarrow 2 \times (1 + S)^A$ , consists of an output function  $o : S \rightarrow 2$  and a transition function  $t : S \rightarrow (1 + S)^A$ . The output function  $o$  indicates whether a state  $s \in S$  is accepting (or terminating) :  $o(s) = 1$ , denoted also by  $s \downarrow$ , or not:  $o(s) = 0$ , denoted by  $s \uparrow$ . The transition function  $t$  associates to each state  $s$  in  $S$  a function  $t(s) : A \rightarrow (1 + S)$ . The set  $1 + S$  is the disjoint union of  $S$  and  $1$ . The meaning of the state transition function is that  $t(s)(a) = \emptyset$  iff  $t(s)(a)$  is undefined, which means that there is no  $a$ -transition from the state  $s \in S$ .  $t(s)(a) \in S$  means that the  $a$ -transition from  $s$  is possible and we define in this case  $t(s)(a) = s_a$ , which is denoted mostly by  $s \xrightarrow{a} s_a$ .

A *bisimulation* between two partial automata  $S = (S, \langle o, t \rangle)$  and  $S' = (S', \langle o', t' \rangle)$  is a relation  $R \subseteq S \times S'$  such that: if  $\langle s, s' \rangle \in R$  then

- (i)  $o(s) = o'(s')$ , i.e.  $s \downarrow$  iff  $s' \downarrow$

- (ii)  $\forall a \in A : s \xrightarrow{a} \Rightarrow (s' \xrightarrow{a} \text{ and } \langle s_a, s'_a \rangle \in R)$ ,  
(iii)  $\forall a \in A : s' \xrightarrow{a} \Rightarrow (s \xrightarrow{a} \text{ and } \langle s_a, s'_a \rangle \in R)$ .

We write  $s \sim s'$  whenever there exists a bisimulation  $R$  with  $\langle s, s' \rangle \in R$ . This relation is the union of all bisimulations, i.e. the greatest bisimulation also called bisimilarity.

### B. Final automaton of partial languages

Below we define the partial automaton of partial languages over an alphabet (input set)  $A$ , denoted by  $\mathcal{L} = (\mathcal{L}, \langle o_{\mathcal{L}}, t_{\mathcal{L}} \rangle)$ . More formally,

$$\mathcal{L} = \{(V, W) \mid V \subseteq W \subseteq A^*, W \neq \emptyset, \text{ and } W \text{ is prefix-closed}\}.$$

The transition function  $t_{\mathcal{L}} : \mathcal{L} \rightarrow (1 + \mathcal{L})^A$  is defined using input derivatives. Recall that for any partial language  $L = (L^1, L^2) \in \mathcal{L}$ ,  $L_a = (L_a^1, L_a^2)$ , where  $L_a^i = \{w \in A^* \mid aw \in L^i\}$ ,  $i = 1, 2$ . If  $a \notin L^2$  then  $L_a$  is undefined. Given any  $L = (L^1, L^2) \in \mathcal{L}$ , the partial automaton structure of  $\mathcal{L}$  is given by:

$$o_{\mathcal{L}}(L) = \begin{cases} 1 & \text{if } \varepsilon \in L^1 \\ 0 & \text{if } \varepsilon \notin L^1 \end{cases}$$

$$t_{\mathcal{L}}(L)(a) = \begin{cases} L_a & \text{if } L_a \text{ is defined} \\ \emptyset & \text{otherwise} \end{cases}$$

Notice that if  $L_a$  is defined, then  $L_a^1 \subseteq L_a^2$ ,  $L_a^2 \neq \emptyset$ , and  $L_a^2$  is prefix-closed. The following notational conventions will be used:  $L \xrightarrow{w} L_w$  iff  $L_w$  is defined iff  $w \in L^2$ .

Recall that  $\mathcal{L} = (\mathcal{L}, \langle o_{\mathcal{L}}, t_{\mathcal{L}} \rangle)$  is final among all partial automata: for any partial automaton  $S = (S, \langle o, t \rangle)$  there exists a unique homomorphism  $l : S \rightarrow \mathcal{L}$ . Another characterization of finality of  $\mathcal{L}$  is that it satisfies the proof principle of coinduction:  $\forall K, L \in \mathcal{L} : K \sim L \Rightarrow K = L$ .

### C. Coinductive definitions

Coinductive definitions as in [8] are used. Recall from [6] the following binary operation on partial languages:

**Definition** (Supremal controllable sublanguage) Define the following binary operation on (partial) languages for all  $K, L \in \mathcal{L}$  and  $\forall a \in A$ :

$$(K /_{A_u c}^{SC} L)_a = \begin{cases} K_a /_{A_u c}^{SC} L_a & \text{if } K \xrightarrow{a} \text{ and } L \xrightarrow{a} \\ & \text{and if } \forall u \in A_u^* : \\ & L_a \xrightarrow{u} \Rightarrow K_a \xrightarrow{u} \\ \emptyset & \text{otherwise} \end{cases}$$

and  $(K /_{A_u c}^{SC} L) \downarrow$  iff  $L \downarrow$ .

We have shown in [6] that for a partial order that considers only second (prefix-closed) components of the languages involved:

**Theorem**  $(K /_{A_u c}^{SC} L) = \sup C(K, L) = \sup \{M \subseteq K : M \text{ is controllable with respect to } L \text{ and } A_u\}$ , i.e.  $K /_{A_u c}^{SC} L$  equals the supremal controllable sublanguage of  $K$ .

### Acknowledgment

Partial financial support of Grant GA AV No. B100190609 and the Academy of Sciences of the Czech Republic, Institutional Research Plan No. AV0Z10190503. is gratefully acknowledged.