Language Equations for Maximal Decompositions in Coordination Control

Komenda, Jan; Lin, Feng; van Schuppen, Jan

DOI
10.1016/j.ifacol.2017.08.2303

Publication date
2017

Document Version
Final published version

Published in
IFAC-PapersOnLine

Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 50-1 (2017) 13441–13446

Language Equations for Maximal Decompositions in Coordination Control

Jan Komenda, * **Feng Lin,** ** **Jan H. van Schuppen** ***

* *Institute of Mathematics, Academy of Sciences of the Czech Republic, Žižkova 22, 616 62 Brno, Czech Republic (e-mail: komenda@math.cas.cz)*
** *Department of Electrical and Computer Engineering, Wayne State University, Detroit, MI, 48098, USA (e-mail: aa0986@wayne.edu)*
*** *Department of Mathematics, Delft University of Technology, Mekelweg 4, 2628 CD Delft, The Netherlands (e-mail: jan.h.van.schuppen@xs4all.nl)*

Abstract: The problem considered is to construct all solutions of an equation for a tuple of languages. The tuple in synchronous composition should equal a considered language. Of special interest are the maximal solutions with respect to a partial order relation on the set of solutions. The motivation of the problem is coordination control. The approach to the problem is to transform the equation to a fixed-point equation. An algorithm is proposed which constructs tuples of maximal solutions by convergence of a sequence of tuples to a solution of the fixed-point equation. Modular supervisory control is shown to benefit from the proposed approach.

Keywords: Discrete-event system, coordination control, convergence, automata, supervisory control.

1. INTRODUCTION

Discrete-event systems Cassandras and Lafortune (2008); Wonham (2008) stemming from modeling human-made systems are often represented by a synchronous product of finite-state machines called automata or generators. The main issue in the modeling and control of such systems is the combinatorial explosion of the number of states, known to be exponential in the number of components in the worst case. In order to alleviate this problem, modular and coordination control architectures have been proposed for supervisory control of such concurrent automata, cf. Ramadge and Wonham (1989); Willner and Heymann (1991); Komenda et al. (2015a).

Coordination control is based on the concept of conditional independence, which then enables one to reduce the coordination control problem (at least in its relaxed form) to modular supervisory control with conditionally decomposable specification. However, we have used so far the infimal (smallest) decomposition as local specification, which is just the global specification projected to the event sets of the local supervisors. Unfortunately, this makes our earlier solutions (local supervisors) restrictive in many cases, because a shared controllable event that needs to be disabled is actually disabled by all local supervisors that have this event in their event sets, unlike decentralized supervisory control of monolithic plants, where the supervisors cooperate and only one or a few disable a shared controllable event.

In this paper, we investigate the language decomposability problem, where canonical minimal solutions are well known Willner and Heymann (1991), but we study rather maximal solutions that are more relevant for modular supervisory control problems. Note that the problem of finding a decomposable sublanguage is undecidable Lin et al. (2014) for general distribution of the global alphabet into modular alphabets. The concept of conditional decomposability Komenda et al. (2012) proceeds by extending locally observable events by coordinator events (communication via the coordinator), which makes every language conditionally decomposable, i.e. decomposable with respect to local alphabets enriched by the coordinator events.

The language decomposability problem is first reformulated as a fixed-point equation, which does not have a supremal solution. We then propose an algorithm that computes a special form of maximal solutions and prove the convergence of the fixed-point iteration scheme to this maximal solution in one step (for each component).

Finally, an application to modular supervisory control is shown, where the computed maximal solutions to the language decomposability problem yield more permissive supervisors. The main advantage of the approach based on conditional decomposability compared to the one based on Nash equilibria Overkamp and van Schuppen (2000) is that the convergence to a maximal solution is established, from which then supervisors can be derived. The approach of this conference paper will be further developed by the authors. The proofs are omitted and will be included in a future paper.

2. MAXIMAL DECOMPOSITION PROBLEM

The reader is assumed to be familiar with the elementary concepts and results of supervisory control of which a short summary is provided in Appendix A.

Call a triple of generators a *coordinated discrete-event system* with the notation, (G_1, G_2, G_k), if the event set of the coordinator contains the intersection of the event set of the tuple of the subsystems G_1 and G_2, that is, if $E_1 \cap E_2 \subseteq E_k$. Let $E = E_1 \cup E_2 \cup E_k$. A language $K \subseteq E^*$ is called *conditionally*

* The research was supported by GAČR grant S15-2532, by RVO 67985840, and by the National Science Foundation of the USA grant 1507096.

10.1016/j.ifacol.2017.08.2303

decomposable with respect to the coordinated discrete-event system (G_1, G_2, G_k) if

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

where $P_{i+k} : E^* \to (E_i \cup E_k)^*$, $i = 1,2$. Further details can be found in Appendix A.

Problem 1. (The *Decomposition Problem* and the *Maximal Decomposition Problem*).

For a coordinated discrete-event system (G_1, G_2, G_k) and a conditionally decomposable sublanguage $K \subseteq E^*$, determine:

(a) all tuples of languages $L_1 \subseteq E^*_{1+k}$ and $L_2 \subseteq E^*_{2+k}$ such that

$$K = L_1 || L_2; \quad (2)$$

Such a tuple will be called a *decomposition* or a *solution* of the decomposition problem;

(b) all maximal tuples of languages which by definition are maximal elements with respect to the partial order relation (see Definition 10). Call such a tuple a *maximal tuple* or a *maximal solution*.

An extension of Problem 1 is to impose additional conditions that the languages of the decomposition are both controllable and observable. This extension is not discussed here due to space limitations. Problem 1 is an extension of the approach to compute supremal sublanguages and to compute controllable subspaces for monolithic DES, Zad et al. (1999).

The motivation for investigating this problem is as follows. Consider a modular generator $G = G_1 || G_2$. Given a prefix-closed specification $\overline{K} = K \subseteq L(G)$, we can always find a coordinator G_k over E_k with $E_1 \cap E_2 \subseteq E_k \subseteq E_1 \cup E_2 = E$ such that (1) $G_k = P_k(G_1) \parallel P_k(G_2)$, and (2) K is conditionally decomposable, that is, $K = P_{1+k}(K) || P_{2+k}(K)$.

Note that: (1) In the worst case, we can take $E_k = E$. (2) An algorithm to find a minimal subset E_k is given in Komenda et al. (2012). (3) $G_k = P_k(G_1) || P_k(G_2)$ implies $G = G_1 || G_2 = G_1 \parallel G_2 \parallel G_k$.

Given a coordinated discrete-event system G_1, G_2, G_k and conditionally decomposable K, the Coordination Control Problem is to find two local supervisors S_i, $i = 1,2$ such that

(1) $L(S_1/[G_1 \| G_k]) \subseteq P_{1+k}(K)$,
(2) $L(S_2/[G_2 \| G_k]) \subseteq P_{2+k}(K)$,
(3) $L(S_1/[G_1 \| G_k]) \parallel L(S_2/[G_2 \| G_k]) = K$.

To solve the above Coordination Control Problem, let us recall the following two definitions from Komenda et al. (2015b).

Definition 1. (Relaxed Conditional Controllability) Given a coordinated discrete-event system G_1, G_2, G_k and conditionally decomposable K, K is *relaxed conditionally controllable* if the projected language $P_{i+k}(K)$ is controllable with respect to $L(G_i||G_k)$ and $E_{i+k,u}$, where $E_{i+k,u} = (E_i \cup E_k) \cap E_u$, for $i = 1,2$.

Definition 2. (Conditional Observability) Given a coordinated discrete-event system G_1, G_2, G_k and conditionally decomposable K, K is called *conditionally observable* if the projected language $P_{i+k}(K)$ is observable with respect to $L(G_i||G_k)$, $E_{i+k,o}$, where $E_{i+k,o} = (E_i \cup E_k) \cap E_o$, for $i = 1,2$.

The following theorem on the existential results of the Coordination Control Problem can then be proved as in Komenda et al. (2015a).

Theorem 3. (Existence Condition) Given a coordinated discrete-event system G_1, G_2, G_k and conditionally decomposable K, there exist supervisors S_1 and S_2 which solve the Coordination Control Problem if and only if

(1) K is relaxed conditionally controllable; and
(2) K is conditionally observable.

If the above necessary and sufficient condition is not satisfied, which is often the case in practice, then our goal is to find supervisors S_1 and S_2 that will achieve the largest sublanguage of K. We use the following Algorithm.

Algorithm 1. Consider a coordinated discrete-event system G_1, G_2, G_k and conditionally decomposable K.

(1) Find the largest languages $L_1 \subseteq E^*_{1+k}$ and $L_2 \subseteq E^*_{2+k}$ such that
$$K = L_1 || L_2.$$
See Algorithm 2 for details.
(2) For $i = 1,2$, find a maximal controllable (with respect to $L(G_i||G_k)$ and $E_{i+k,u}$) and observable (with respect to $L(G_i||G_k)$, $E_{i+k,o}$) sublanguage of $L_i \cap L(G_i||G_k)$, denoted by $L_i^{\uparrow}$.
(3) For $i = 1,2$, synthesize a supervisor S_i for $G_i|||G_k$ such that $L(S_i/[G_i||G_k]) = L_i^{\uparrow}$.

Algorithm 1 gives local supervisors greatest freedom to generate the largest languages while ensuring that the specification K is satisfied, which is shown below.

Theorem 4. Given a coordinated discrete-event system G_1, G_2, G_k and conditionally decomposable K, the supervisors S_1 and S_2 obtained by Algorithm 1 have the following property,

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

Since Steps 2 and 3 of Algorithm 1 are well understood, in this paper, we will investigate how to do Step 1 of the Algorithm.

Note that although $L_i \subseteq E^*_{i+k}$, when we find its maximal controllable and observable sublanguage, L_i is automatically intersected with $L(G_i||G_k)$, hence $L_i^{\uparrow} \subseteq L(G_i \| G_k)$.

Example 1. Let us consider the following simple example. $K = \overline{\{abc, bac\}}$ and consider its decomposition into event sets $E_{1+k} = \{a, c\}$ and $E_{2+k} = \{b, c\}$. The canonical (infimal) decomposition is $P_{1+k}(K) = \overline{\{ac\}}$ and $P_{2+k}(K) = \overline{\{bc\}}$. However, it is obvious that the following decompositions tuples are also possible: $(L_1, L_2) = (\overline{\{ac^*\}}, \overline{\{bc\}}$ and $(L_1', L_2') = (\overline{\{ac\}}, \overline{\{bc^*\}})$. These decompositions are maximal with respect to the order on tuples of languages (see Definition 10). These tuples are incomparable and a supremal decomposition does not exist.

3. EQUATIONS OF LANGUAGES

Our approach to solve the Maximal Decomposition Problem consists of two steps. In Step 1, we convert the problem of finding languages $L_1 \subseteq E^*_{1+k}$ and $L_2 \subseteq E^*_{2+k}$, i.e. the tuple $(L_1, L_2) \in \mathrm{Pwr}(E^*_{1+k}) \times \mathrm{Pwr}(E^*_{2+k})$ such that $K = L_1 || L_2$, into a problem of finding the fixed point of some language equations. Note that theory of fixed points stems from the lattice framework, cf. Tarski (1955), and formal languages form a lattice structure. In Step 2, we define a sequence of tuples with special initial conditions and show that the sequence converges to the fixed point of the equations of languages and that the

fixed point is maximal. To this end, let us define, for a tuple $(L_1,L_2) \in \mathrm{Pwr}(E^*_{1+k}) \times \mathrm{Pwr}(E^*_{2+k})$, its *asymmetric equations* as,

$$L_1 = L_1||P_k(L_2), \tag{3}$$
$$L_2 = P_k(L_1)\|L_2. \tag{4}$$

The following set theoretic lemma is needed. Given a universal set U that contains all sets. Let $A \subseteq U$ and let $A^c = U \setminus A$ denote the complement of A with respect to U.

Lemma 5. Consider sets $A, B \subseteq U$. Assume $B \subseteq A$. A set $X \subseteq U$ is a solution of the equation $B = X \cap A$ if and only if X is a solution of the fixed-point equation,

$$X = B \cup (X \cap A^c).$$

To solve the Maximal Decomposition Problem, we first prove the following proposition.

Proposition 6. Consider a coordinated discrete-event system $(G_1,\ G_2,\ G_k)$ and a conditionally decomposable K. For any $(L_1,L_2) \in \mathrm{Pwr}(E^*_{1+k}) \times \mathrm{Pwr}(E^*_{2+k})$, $K = L_1||L_2$ if and only if

$$P_{1+k}(K) = L_1||P_{1+k}(L_2) = L_1 \cap (P_k^{1+k})^{-1}P_k(L_2),$$
$$\wedge\ P_{2+k}(K) = P_{2+k}(L_1)||L_2 = (P_k^{2+k})^{-1}P_k(L_1) \cap L_2.$$

Now we can prove the following theorem.

Theorem 7. Consider a coordinated discrete-event system G_1, G_2, G_k and conditionally decomposable K. For any $(L_1,L_2) \in \mathrm{Pwr}(E^*_{1+k}) \times \mathrm{Pwr}(E^*_{2+k})$, $K = L_1||L_2$ if and only if (L_1,L_2) is a solution of the following fixed-point equations,

$$L_1 = P_{1+k}(K) \cup \left(L_1 \cap ((P_k^{1+k})^{-1}P_k(L_2))^c\right), \tag{5}$$
$$L_2 = P_{2+k}(K) \cup \left(L_2 \cap ((P_k^{2+k})^{-1}P_k(L_1))^c\right). \tag{6}$$

The operator introduced next will be useful for the remainder of the paper.

Definition 8. For a tuple $(L_1,L_2) \in \mathrm{Pwr}(E^*_{1+k}) \times \mathrm{Pwr}(E^*_{2+k})$, define the *language operator* H as,

$$H : (\mathrm{Pwr}(E^*_{1+k}) \times \mathrm{Pwr}(E^*_{2+k})) \to (\mathrm{Pwr}(E^*_{1+k}) \times \mathrm{Pwr}(E^*_{2+k})),$$
$$(L_1,L_2) \mapsto H(L_1,L_2) = (H_1(L_1,L_2), H_2(L_1,L_2),$$
$$H_1(L_1,L_2) = P_{1+k}(K) \cup \left(L_1 \cap ((P_k^{1+k})^{-1}P_k(L_2))^c\right), \tag{7}$$
$$H_2(L_1,L_2) = P_{2+k}(K) \cup \left(L_2 \cap ((P_k^{2+k})^{-1}P_k(L_1))^c\right). \tag{8}$$

It follows directly from Theorem 7 and the above definition that the tuple (L_1,L_2) is a solution of the decomposition problem if and only if,

$$(L_1,L_2) = H(L_1,L_2) = (H_1(L_1,L_2), H_2(L_1,L_2)). \tag{9}$$

Proposition 9. The language operator defined above has the following monotonicity properties.

(a) H_1 is monotonically increasing with respect to its first variable, that is, for any fixed $L_2 \subseteq E^*_{2+k}$,
$$L_1 \subseteq L_1' \Rightarrow H_1(L_1,L_2) \subseteq H_1(L_1',L_2).$$

(b) H_1 is monotonically decreasing with respect to its second variable, that is, for any $L_1 \subseteq E^*_{1+k}$,
$$L_2 \subseteq L_2' \Rightarrow H_1(L_1,L_2) \supseteq H_1(L_1,L_2').$$

(c) H_2 is monotonically increasing with respect to its second variable, that is, for any $L_1 \subseteq E^*_{1+k}$,
$$L_2 \subseteq L_2' \Rightarrow H_2(L_1,L_2) \subseteq H_2(L_1,L_2').$$

(d) H_2 is monotonically decreasing with respect to its first variable, that is, for any $L_2 \subseteq E^*_{2+k}$,
$$L_1 \subseteq L_1' \Rightarrow H_2(L_1,L_2) \supseteq H_2(L_1',L_2).$$

The approach of this paper is to start with a language operator of the form $H = (H_1,H_2)$ with the monocity properties of Proposition 9 and then to investigate the maximal tuples.

4. ORDERING ON TUPLES OF LANGUAGES

We now formally define the ordering on tuples of languages as follows.

Definition 10. For tuples of languages
$$(L_1,L_2) \in \mathrm{Pwr}(E^*_{1+k}) \times \mathrm{Pwr}(E^*_{2+k}),$$
define the order relation $\leq$ on the set of such tuples as

$$(L_1,L_2) \leq (L_1',L_2'),\ \text{ if }\ L_1 \subseteq L_1' \wedge L_2 \subseteq L_2'. \tag{10}$$

In addition, define the strict order relation $<$ as,

$$(L_1,L_2) < (L_1',L_2'),\ \text{if,}$$
$$(L_1 \subset L_1' \wedge L_2 \subseteq L_2') \vee (L_1 \subseteq L_1' \wedge L_2 \subset L_2'),$$

where $L_1 \subset L_1'$ means strict containment, that is, $L_1 \subset L_1'$ if and only if $L_1 \subseteq L_1' \wedge L_1 \neq L_1'$.

Partial order relations on languages were studied in Fabre and Benveniste (2007).

Recall the concepts of a tuple of the decomposition problem and that of a maximal tuple, see Problem 1, which are defined with respect to the above order relation. We have the following remark, which specifies some special cases that we exclude from our consideration.

Remark 11. (1) It is shown in Willner and Heymann (1991) that if $K = L_1||L_2$ then,
$$P_{i+k}(K) \subseteq L_i,\ \ i = 1,2. \tag{11}$$
Thus, $(L_1,L_2) = (P_{1+k}(K), P_{2+k}(K))$ is minimal tuple satisfying $K = L_1||L_2$. This minimal tuple is unique, and hence it is the infimal tuple.

(2) In the case of $E_{1+k} \cap E_{2+k} = \emptyset$ (pure shuffle), there is a unique solution to $K = L_1\|L_2$, namely the infimal tuple $(L_1,L_2) = (P_{1+k}(K), P_{2+k}(K))$. It is then also the supremal/maximal tuple. This case is not interesting and we will assume from now on that $E_{1+k} \cap E_{2+k} \neq \emptyset$.

(3) In the case of $E_{1+k} = E_{2+k} = E$, there are many tuples satisfying $K = L_1\|L_2$. For example, we can fix one component to $L_1 = P_{1+k}(K) = K$ and the other component has the following form: $L_2 = M$, where $K \subseteq M \subseteq E^*$. The two (incomparable) maximal tuples, among many others, are then (K,E^*) and (E^*,K). This case is not interesting and we will exclude it. The case where either $E_{1+k} \subseteq E_{2+k}$ or $E_{2+k} \subseteq E_{1+k}$ is also not interesting and we will exclude these cases. Moreover, we will assume that $P_{1-k}(K) \neq \emptyset$ and $P_{2-k}(K) \neq \emptyset$, where $P_{i-k} : E^* \to (E_i \setminus E_k)^*$ for $i = 1,2$. The last assumption means that the decomposition problem is nontrivial on both local alphabets.

5. A SEQUENCE OF TUPLES AND THEIR CONVERGENCE

Definition 12. For a tuple $(L_1, L_2) \in \mathrm{Pwr}(E^*_{1+k}) \times \mathrm{Pwr}(E^*_{2+k})$, define the *operator equations* as

$$(L_1, L_2) = H(L_1, L_2) = (H_1(L_1, L_2),\ H_2(L_1, L_2)), \quad (12)$$

$$\Leftrightarrow$$

$$L_1 = H_1(L_1, L_2), \quad (13)$$

$$L_2 = H_2(L_1, L_2). \quad (14)$$

The equations are fixed-point equations of the tuple.

Corollary 13. (of Theorem 7). A tuple of languages is a maximal solution of Problem 1 if and only if it is a maximal solution of the operator equations of Definition 12.

One way to find a solution to the fixed-point equations is to use the following iterative procedure:

Definition 14. Consider the language operator H, define a *sequence of tuples of languages* by,

$$(L_1^{(0)}, L_2^{(0)}) \in \mathrm{Pwr}(E^*_{1+k}) \times \mathrm{Pwr}(E^*_{2+k}),$$

$$\{(L_1^{(i)}, L_2^{(i)}) \in \mathrm{Pwr}(E^*_{1+k}) \times \mathrm{Pwr}(E^*_{2+k}),\ \forall\, i \in \mathbb{N}\},$$

$$L_1^{(i+1)} = H_1(L_1^{(i)}, L_2^{(i)}), \quad (15)$$

$$L_2^{(i+1)} = H_2(L_1^{(i)}, L_2^{(i)}). \quad (16)$$

Of interest is whether the sequence converges and, if so, what the languages of the limit are. The following partial results are useful for the decomposition problem.

Proposition 15. For a sequence of languages obtained above, there holds,

(a) If L_2 is fixed, then
$$H_1(H_1(L_1, L_2), L_2) = H_1(L_1, L_2).$$

(b) If L_1 is fixed, then
$$H_2(L_1, H_2(L_1, L_2)) = H_2(L_1, L_2).$$

Let us now investigate sequences of tuples $\{(L_1^{(i)}, L_2^{(i)}) : i \in \mathbb{N}\}$ with two special initial conditions.

Proposition 16. Consider a coordinated discrete-event system (G_1, G_2, G_k) and a conditionally decomposable language K. The sequences converges in one step to the infimal solution for the following two cases defined by different initial languages.

(a)

$$(L_1^{(0)}, L_2^{(0)}) = (P_{1+k}(K), P_{2+k}(K)), \quad (17)$$

$$(L_1^{(\infty)}, L_2^{(\infty)}) = (L_1^{(1)}, L_2^{(1)}) = (P_{1+k}(K), P_{2+k}(K)), \quad (18)$$

(b)

$$(L_1^{(0)}, L_2^{(0)}) = (E^*_{1+k}, E^*_{2+k}), \quad (19)$$

$$(L_1^{(\infty)}, L_2^{(\infty)}) = (L_1^{(1)}, L_2^{(1)}) = (P_{1+k}(K), P_{2+k}(K)). \quad (20)$$

The above results show that the the domain of attraction of sequences of languages is rather irregular. Starting from either the smallest or the largest initial languages, the sequence does not converge to a maximal solution of the fixed-point equation.

6. CONSTRUCTION OF A MAXIMAL SOLUTION

In this section, we formally propose a solution to the maximal decomposition problem using the following algorithm.

Algorithm 2. Construction of a maximal tuple of languages. Consider a coordinated discrete-event system (G_1, G_2, G_k) and a conditionally decomposable sublanguage K.

(1) Fix $(L_1^{(0)}, L_2^{(0)}) = (P_{1+k}(K), E^*_{2+k})$.
(2) Compute $L_2^{(1)} = H_2(L_1^{(0)}, E^*_{2+k})$.
(3) Compute $L_1^{(1)} = H_1(E^*_{1+k}, L_2^{(1)})$.

Set $(L_1^{(*)}, L_2^{(*)}) = (L_1^{(1)}, L_2^{(1)})$.

The tuple of languages of the limit depends on the initial tuple of languages. The limit of the initial tuple $(L_1^{(0)}, L_2^{(0)}) = (E^*_{1+k}, P_{2+k}(K))$ is different from that of Proposition 16. As a preliminary result to the proof that the above algorithm yields a maximal tuple, a technical lemma and a convergence theorem are presented below. Further investigation is needed into the time-complexity when the algorithm is applied to regular languages.

Lemma 17. For a natural projection $P_k : E^* \to E_k^*$ and any language $L \subseteq E^*$ the following inclusion holds for natural projection of the complement of L:

$$P_k(L^c) \supseteq (P_k(L))^c \quad (21)$$

Theorem 18.

(a) In Algorithm 2, there is no change in L_1,
$$L_1^{(*)} = L_1^{(0)} = P_{1+k}(K).$$

(b) The tuple $(L_1^{(*)}, L_2^{(*)})$ is a fixed-point of the language equations,
$$L_1^{(*)} = H_1(L_1^{(*)}, L_2^{(*)})$$
$$L_2^{(*)} = H_2(L_1^{(*)}, L_2^{(*)}).$$

Now that $(L_1^{(*)}, L_2^{(*)})$ is a solution of the fixed-point equations, we want to show that it is a maximal solution. We need the following Lemma.

Lemma 19. Define the event set $E_{1-k} = E_1 \setminus E_k$ and assume that $P_k(K) \neq \emptyset$.
If $(P_k^{1+k})^{-1}(P_k(K)) = P_{1+k}(K)$ then $P_{1-k}(K) = E^*_{1-k}$.

Theorem 20. The language tuple $(L_1^{(*)}, L_2^{(*)})$ produced by Algorithm 2 is a maximal tuple.

It should be clear that by monotonicity of projections and inverse projections, any language tuple of the form $(L_1^{(*)}, L_2)$ = $(P_{1+k}(K), L_2)$, where $P_{2+k}(K) \subseteq L_2 \subseteq L_2^{(*)}$ is also a solution of the language decomposability problem.

Finally, we note that by interchanging indices in Algorithm 2 we can obtain a second maximal solution.

7. NASH EQUILIBRIUM

In a dynamic game problem there are two controllers or players, see Basar and Olsder (1999), each controller chooses his control law so as to maximize its cost function. The controllers do not communicate about their choices. For this reason the tuple of controllers aims for a Nash equilibrium defined below.

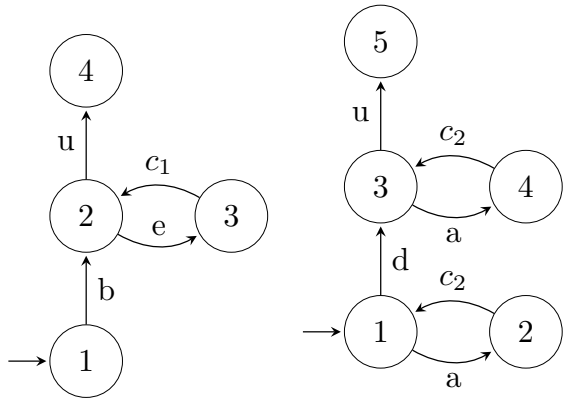

Fig. 1. Generators G_1 (left) and G_2 (right).

Definition 21. The tuple $(L_1^{(*)}, L_2^{(*)}) \in (\mathrm{Pwr}E_{1+k}^* \times \mathrm{Pwr}E_{2+k}^*)$ is a *Nash equilibrium* for the setting if

$$K = L_1^{(*)} \| L_2^{(*)},$$
$$\Big((\forall\, L_1 \subseteq E_{1+k}^*)\; K = L_1 \| L_2^{(*)}\Big) \Rightarrow (L_1, L_2^{(*)}) \leq (L_1^{(*)}, L_2^{(*)}),$$
$$\Big((\forall\, L_2 \subseteq E_{2+k}^*)\; K = L_1^{(*)} \| L_2\Big) \Rightarrow (L_1^{(*)}, L_2) \leq (L_1^{(*)}, L_2^{(*)}).$$

A maximal tuple is then a Nash equilibrium but the converse is in general not true. In case of a decentralized control problem, treated in Overkamp and van Schuppen (2000) both controllers have the same cost criterion. The problem of determining a Nash equilibrium as defined above is thus of interest.

Theorem 22. The tuple of languages $(L_1^{(*)}, L_2^{(*)}) \in (\mathrm{Pwr}E_{1+k}^* \times \mathrm{Pwr}E_{2+k}^*)$ produced by Algorithm 2 is a Nash equilibrium.

A procedure to construct Nash equilibria as defined in Definition 14 for a sequence of sublanguages, was provided in Overkamp and van Schuppen (2000). There exists an example where the sequence does not converge while a Nash equilibrium exists, (Overkamp and van Schuppen, 2000, Example 4.6).

8. APPLICATION TO SUPERVISORY CONTROL WITH COORDINATION

In this section, an example relevant for application of our decomposability problem in supervisory control is presented.

Example 2. Let us consider two generators G_1 and G_2 over event sets $E_1 = E_{1+k} = \{b, e, c_1, u\}$ and $E_2 = E_{2+k} = \{a, d, c_2, u\}$ respectively, displayed in Fig. 1. Let $E_u = E_k = \{u\}$ and $G = G_1 \| G_2$.

The specification K is obtained from G by removing the forbidden states $(4,5)$ in $G = G_1 \| G_2$, that is, the uncontrollable event u is not allowed.

For comparison, we first consider our previous approach to modular/coordination control, where local specifications correspond to infimal decompositions of K, that is, $P_{i+k}(K)$, $i = 1,2$. In the example, $P_{1+k}(K)$ and $P_{2+k}(K)$ correspond to the automata obtained from G_1 and G_2 by removing the top states 4 and 5 respectively. The corresponding local controllers then need to disable events b in G_1 and d in G_2. The resulting closed-loop is however restrictive, because the local closed-loop for G_2 only contains the states 1 and 2, while event d is disabled. We will show that in the solution proposed in this paper, we do not need to disable the event d. Indeed, if we apply the results and algorithms in this paper, we find much richer overall closed-loop behaviors.

First, we apply Algorithm 2 to find a maximal decomposition of K as follows.

(1) Fix $L_1^{(0)} = P_{1+k}(K) = \overline{b(ec_1)^*}$.
(2) Compute

$$\begin{aligned} L_2^{(*)} \cap L(G_2) &= H_2(L_1^{(0)}, E_{2+k}^*) \cap L(G_2) \\ &= (P_{2+k}(K) \cup \Big((P_k^{2+k})^{-1} P_k(K)\Big)^c) \cap L(G_2) \\ &= (P_{2+k}(K) \cup E_{2+k}^* u E_{2+k}^*) \cap L(G_2) = L(G_2). \end{aligned}$$

(3) We know from Theorem 18 $L_1^{(*)} = H_1(E_{1+k}^*, L_2^{(*)}) = P_{1+k}(K)$.

We see that there is no need to control $L(G_2)$ at all, because $(P_{1+k}(K), L(G_2))$ is a (maximal) solution to the decomposability problem. It is sufficient to use a supervisor for $L(G_1)$ to ensure that the closed-loop local language stays within $P_{1+k}(K)$. Then the overall closed-loop language using Algorithm 1 is given by

$$L(S_1/[G_1 \| G_k]) || L(S_2/[G_2 \| G_k]) = {L_1^{(*)}}^{\uparrow} || {L_2^{(*)}}^{\uparrow},$$

where ${L_1^{(*)}}^{\uparrow} = \{\varepsilon\}$, because b has to be disabled by S_1 to achieve the local specification $L_1^{(*)} = P_{1+k}(K)$; and ${L_2^{(*)}}^{\uparrow} = L(G_2)$, because the local specification $L_2^{(*)} = L(G_2)$. Note that the solution (the closed-loop language) based on the maximal decomposition $(L_1^{(*)}, L_2^{(*)})$ is a strictly larger sublanguage of K than the solution based on the infimal decomposition $(P_{1+k}(K), P_{2+k}(K))$, namely S_2 does not need to disable the events d and c_2 in the solution we have proposed, hence both states 3 and 4 of G_2 are kept.

Note that we have chosen to set initially $L_1^{(0)} = P_{1+k}(K)$. The symmetric maximal solution can be obtained by setting $L_2^{(0)} = P_{2+k}(K) = \overline{(ac_2)^* d (ac_2)^*}$. Then according to Algorithm 2 we compute

$$\begin{aligned} L_1^{(*)} \cap L(G_1) &= H_1(E_{1+k}^*, L_2^{(0)}) \cap L(G_1) \\ &= (P_{1+k}(K) \cup \Big((P_k^{1+k})^{-1} P_k(K)\Big)^c) \cap L(G_1) \\ &= (P_{1+k}(K) \cup E_{2+k}^* u E_{2+k}^*) \cap L(G_1) = L(G_1). \end{aligned}$$

Finally, we know from the convergence result of Theorem 18 that $L_2^{(*)} = H_2(L_1^{(*)}, E_{2+k}^*) = P_{2+k}(K)$. We see that there is no need to control $L(G_1)$ at all, because $(L(G_1), P_{2+k}(K))$ is a (maximal) solution to the decomposability problem. It is sufficient to use a supervisor for $L(G_2)$ to ensure that the closed-loop local language stays within $P_{2+k}(K)$. Therefore, in this symmetric maximal solution we only need to disable d, but b need not be disabled. Clearly, both symmetric solutions (the corresponding closed-loops) are not comparable and at the same time both solutions are more permissive than those obtained in the original coordination control approach based on the infimal decomposition of the specification.

9. CONCLUDING REMARKS

We have studied language decomposability problem, maximal solutions have been proposed and computed in a systematic

way based on fixed-point characterization of solutions. We have shown the convergence of the fixed-point iteration scheme to this maximal solution. Our solution has been applied to modular supervisory control, where more permissive supervisors are obtained compared to our previous approach. The solution will also be useful for decentralized control problems, where larger coobservable sublanguages can be obtained.

REFERENCES

Basar, T. and Olsder, G.J. (1999). *Dynamic noncooperative game theory (2nd Ed.)*. Number 23 in Classics in Applied Mathematics. SIAM, Philadelphia.

Cassandras, C.G. and Lafortune, S. (2008). *Introduction to discrete event systems*. Springer, New York, 2nd edition.

Fabre, E. and Benveniste, A. (2007). Partial order techniques for distributed discrete event systems: Why you cannot avoid using them. *Discrete Event Dynamics Systems*, 17, 355–403. doi:s10626-007-0016-1.

Komenda, J., Masopust, T., and van Schuppen, J.H. (2012). On conditional decomposability. *Systems & Control Lett.*, 61, 1260–1268. doi:10.1016/j.sysconle.2012.07.013.

Komenda, J., Masopust, T., and van Schuppen, J.H. (2015a). Coordination control of discrete-event systems revisited. *Discrete Event Dynamics Systems*, 25, 65–94. doi: 10.1007/s10626-013-0179-x.

Komenda, J., Masopust, T., and van Schuppen, J.H. (2015b). A relaxed framework for coordination control of discrete-event systems. Report arXiv:1501.07859v1, arXiv.

Lin, F. and Wonham, W. (1988). On observability of discrete-event systems. *Information Sciences*, 44, 173–198.

Lin, L., Stefanescu, A., Su, R., Wang, W., and Shehabinia, A.R. (2014). Towards decentralized synthesis: Decomposable sublanguage and joint observability problems. In *2014 American Control Conference (ACC.2014)*, 2047–2052. IEEE.

Overkamp, A. and van Schuppen, J.H. (2000). Maximal solutions in decentralized supervisory control. *SIAM J. Control & Opt.*, 39, 492–511.

Ramadge, P. and Wonham, W. (1987). Supervisory control of a class of discrete event processes. *SIAM J. Control & Opt.*, 25, 206–230.

Ramadge, P. and Wonham, W. (1989). The control of discrete event systems. *Proc. IEEE*, 77, 81–98.

Tarski, A. (1955). A lattice-theoretical fixpoint theorem and its applications. *Pacific J. Math.*, 5, 285–309.

Willner, Y. and Heymann, M. (1991). Supervisory control of concurrent discrete-event systems. *Int. J. Control*, 54, 1143–1169.

Wonham, W. (2008). *Supervisory control of discrete-event systems*. Publisher W.M. Wonham, Toronto.

Zad, S.H., Kwong, R., and Wonham, W. (1999). Supremum operators and computation of supremal elements in system theory. *SIAM J. Control & Opt.*, 37, 695–709.

Appendix A. SUPERVISORY CONTROL

Let us briefly recall some basic concepts and notations from supervisory control theory of discrete-event systems. We use a generator (finite-state machine or automaton) to model a discrete-event system. A *generator* is a quadruple

$$G = (Q, E, \delta, q_0),$$

consisting of a finite set of *states* Q, a finite *event set* E, a *partial transition function* $\delta : Q \times E \to Q$, and the *initial state* $q_0 \in Q$. The transition function δ can be extended in the standard way to strings, that is, $\delta : Q \times E^* \to Q$, where E^* is the set of finite sequences (words) over alphabet E. A language is a set of words, i.e. $L \subseteq E^*$. We also write it as $L \in \mathrm{Pwr}(E^*)$, where Pwr is the notation for the powerset (the set of all subsets). We only study prefix-closed languages in this paper, hence only the *language generated* by G is considered. which is defined as

$$L(G) = \{s \in E^* \mid \delta(q_0, s) \in Q\}.$$

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

A prefix-closed language $K \subseteq L(G)$ is said to be *controllable* with respect to $L(G)$ and E_u if Ramadge and Wonham (1987)

$$KE_u \cap L(G) \subseteq K.$$

A *natural projection* $P : E^* \to D^*$, for $D \subseteq E$, is a homomorphism defined as $P(a) = \varepsilon$, for $a \in E \setminus D$, and $P(a) = a$, for $a \in D$. The *inverse image* of P, denoted by $P^{-1} : D^* \to Pwr(E^*)$, is defined as $P^{-1}(w) = \{s \in E^* \mid P(s) = w\}$. These definitions can be extended to languages. For a generator G and a projection P, $P(G)$ denotes the minimal generator such that $L(P(G)) = P(L(G))$. The reader is referred to Cassandras and Lafortune (2008); Wonham (2008) for a construction of $P(G)$. A generator G is said to be partially observed if only a proper subset of events $E_o \subset E$, called the set of *observable events*, is observed. Then partial observations of strings are given by natural projection $P : E^* \to E_o^*$ as defined above with $D = E_o$. A prefix-closed language $K \subseteq L(G)$ is said to be *observable* with respect to $L(G)$ and E_o if Lin and Wonham (1988)

$$(\forall s, s' \in K)(\forall a \in E)$$
$$((P(s) = P(s') \wedge sa \in K \wedge s'a \in L(G)) \Rightarrow s'a \in K).$$

A (partial observation) *supervisor* for a controlled generator (G, E_c, Γ) is a map $S : P(L(G)) \to \Gamma$. The language of the *closed-loop system* associated with controlled generator (G, E_c, Γ) and supervisor S is defined as the smallest language $L(S/G)$ such that (1) $\varepsilon \in L(S/G)$, and (2) for any $s \in L(S/G)$, $sa \in L(G)$ and $a \in S(P(s))$ implies $sa \in L(S/G)$.

Given a control specification represented by a prefix-closed language $K \subseteq L(G)$, the aim of supervisory control with partial observations is to find a supervisor S such that $L(S/G) = K$. Such a supervisor exists if and only if K is controllable with respect to $L(G)$ and E_u and observable with respect to $L(G)$ and E_o Lin and Wonham (1988).

The above supervisor is a centralized supervisor for a monolithic model of G. Such a centralized control may not be adequate for complex systems that are often encountered in practice. To reduce computational complexity, coordination control is proposed Komenda et al. (2015a). G is often obtained by synchronous product of several generators $G_i, i = 1, 2, ..., n$:

$$G = G_1 || G_2 || ... || G_n,$$

where $G_i = (Q_i, E_i, \delta_i, q_{0,i})$ has event set (local event set) $E_i \subseteq E$. For alphabets E_i, E_j, $E_\ell \subseteq E$, P_ℓ^{i+j} denotes the projection from $(E_i \cup E_j)^*$ to E_ℓ^*. If $E_i \cup E_j = E$, we simply write P_ℓ.

The synchronous product of languages $L_i \subseteq E_i^*$, $i = 1, \ldots, n$, is defined as $\|_{i=1}^n L_i = \cap_{i=1}^n P_i^{-1}(L_i) \subseteq E^*$, where $E = \cup_{i=1}^n E_i$ and $P_i : E^* \to E_i^*$ are projections to local event sets. It is known that $L(\|_{i=1}^n G_i) = \|_{i=1}^n L(G_i)$ (see Cassandras and Lafortune (2008) for more details).