INSTITUT DE RECHERCHE EN INFORMATIQUE ET SYSTÈMES ALÉATOIRES

PUBLICATION INTERNE

N° 1593

SUPERVISORY CONTROL OF CONCURRENT DISCRETE EVENT SYSTEMS

BENOIT GAUDIN, HERVÉ MARCHAND

ISSN 1166-8687

IRISA
CAMPUS UNIVERSITAIRE DE BEAULIEU - 35042 RENNES CEDEX - FRANCE

INSTITUT DE RECHERCHE EN INFORMATIQUE ET SYSTÈMES ALÉATOIRES
Campus de Beaulieu – 35042 Rennes Cedex – France
Tél. : (33) 02 99 84 71 00 – Fax : (33) 02 99 84 71 71
http://www.irisa.fr

Supervisory Control of Concurrent Discrete Event Systems

Benoit Gaudin, Hervé Marchand

Systèmes communicants
Projet VerTeCs

Publication interne n1593 — February 2004 — 16 pages

Abstract: In this paper, we are interested in the control of a particular class of Concurrent Discrete Event Systems defined by a collection of components that interact with each other. We investigate the computation of the supremal controllable language contained in the one of the specification. We do not adopt the decentralized approach. Instead, we have chosen to perform the control on some approximations of the plant derived from the behavior of each component. The behavior of these approximations is restricted so that they respect a new language property for discrete event systems called *partial controllability condition* that depends on the specification. It is shown that, under some assumptions, the intersection of these "controlled approximations" corresponds to the supremal controllable language contained in the specification with respect to the plant. This computation is performed without having to build the whole plant, hence avoiding the state space explosion induced by the concurrent nature of the plant.

Key-words: Discrete Event Systems, Supervisory Control Problem, Concurrent Finite State Machines, Partial controllability

(Résumé : tsvp)

Centre National de la Recherche Scientifique
(UMR 6074) Université de Rennes 1 – Insa de Rennes

Institut National de Recherche en Informatique
et en Automatique – unité de recherche de Rennes

Contrôle de systèmes à événements discrets concurrents

Résumé : Dans ce rapport, nous nous intéressons au contrôle de systèmes à événements discrets concurrents définis par une collection de sous-systèmes interagissant les uns avec les autres. Étant donné un objectif de contrôle, le but consiste à calculer un superviseur maximal (i.e le plus permissif) assurant cet objectif, sans construire explicitement le système à contrôler. Des approximations du système G sont dérivés à partir des sous-systèmes qui le composent, et une propriété appelée *contrôlabilité partielle*, devant être vérifiée par l'objectif de contrôle sur ces approximations, est introduite. Assurer la contrôlabilité partielle de l'objectif de contrôle sur chacune des approximations permet, sous certaines hypothèses, d'en déduire un superviseur maximal assurant l'objectif de contrôle sur G. De plus, les calculs effectués ont une faible complexité et ne nécessitent pas de construire explicitement le système G, évitant ainsi l'explosion combinatoire inhérent aux systèmes concurrents.

Mots clés : Systèmes à événements discrets, contrôle, systèmes concurrents , contrôlabilité partielle

1 Introduction

In this paper, we are interested in the control of Concurrent Discrete Event Systems defined by a collection of components that interact with each other. *Supervisory control* [8] consists in modifying a system (plant) such that the modified (or *controlled*) system satisfies a given property. Typically, the behaviors of the controlled system are constrained to remain among those allowed by a formal specification. We adopt the formalism of Supervisory Control theory [8] and model the system as the regular language generated by a Finite State Machine (FSM). Given a plant and a specification modeled by languages, in the Ramadge & Wonham [8] theory, one important phase is the computation of the supremal controllable sub-language contained in a language that represents the expected behavior. However, although this computation is polynomial in the number of states of the plant and of the specification, it is well known that the plant size grows exponentially with the number of components that compose the plant. This renders the computation of the supervisors not practical because of the size of the generated state space which is often too important when dealing with large scale systems.

Several approaches have been recently investigated to deal with the complexity issue of the control of Concurrent Discrete Event Systems. Given a Concurrent Discrete Event System $G = G_1 \parallel \cdots \parallel G_n$ and a specification expressed by a language K, the problem is to compute the supremal controllable sublanguage of $K \cap \mathcal{L}(G)$ w.r.t. $\mathcal{L}(G)$ without having to build $\mathcal{L}(G)$. In [3, 6], the authors considers the control of a product plant (i.e. systems composed of asynchronous subsystems, not sharing common events). Given a specification, a local system is built from the components that are coordinated by the specification (i.e. all the components that share some events used to express it). It is then sufficient to compute the local supervisor ensuring the specification with respect to this local system in order to obtain the result on the whole system. Closely related to the decentralized theory, under the hypothesis that the specification is separable and that the shared events are controllable, the authors of [11] provide a solution allowing to compute local modular supervisors $\mathcal{S}_i$ acting upon G_i and to operate the individually controlled system $\mathcal{S}_i/G_i$ concurrently in such a way that the behavior of controlled plant (i.e. $\mathcal{L}(\|_i\ \mathcal{S}_i/G_i)$) corresponds to the supremal controllable sublanguage of K w.r.t. the plant $\mathcal{L}(G)$. The same methodology has been used in [9] for the control of concurrent plant for which the various components have an identical structure. Knowing that the local supervisors $\mathcal{S}_i$ are only operating on a subset of the local events, they give necessary and sufficient conditions on the specification K to obtain a non-blocking controlled plant that exactly matches the specification K. See also [1, 5, 7] for other works relating to the modular control of concurrent plant.

In this paper, compared to [11], we adopt a dual approach. Instead of having one local supervisor per component that enforces local control actions with respect to the events of this component, we have chosen to perform the control on some approximations of the plant derived from the behavior of each component. The behavior of these abstractions is resctricted so that they respect a new language property for discrete event systems called *partial controllability condition* that depends on K. Under some assumptions on the specification, it is shown that a supervisor can be derived from these "controlled approximations" such that the behavior of the controlled plant corresponds to the supremal controllable language contained in K with respect to the plant G.

The rest of the paper is as follows: in Section 2, the model and the main concept of Supervisory Control are introduced. In Section 3, the notion of partial controllability is defined. Finally, the fourth Section is dedicated to the modular synthesis of the supremal controllable language of a specification w.r.t. a concurrent DES.

2 Preliminaries

2.1 Model and Supervisory Control overview

The basic structures from which the plant is built are Finite State Machines (FSM)

Definition 1 *An FSM is defined by a 4-tuple $G = \langle \Sigma, \mathcal{X}, x_o, \delta \rangle$, where Σ is the finite alphabet of G. $\mathcal{X}$ is the finite set of states, $x_o \in \mathcal{X}$ is the initial state, whereas δ is the partial transition function defined over $\Sigma \times \mathcal{X} \longrightarrow \mathcal{X}$.*

The notation $\delta(\sigma, x)!$ means that $\delta(\sigma, x)$ is defined, i.e., there is a transition labeled by an event σ out of state x in machine G. Likewise, for $x \in \mathcal{X}$ and $s \in \Sigma^*$, $\delta(s, x)$ denotes the state reached by taking the sequence of events defined by trace s from state x in machine G. The behavior of the system is described by the language $\mathcal{L}(G) \subseteq \Sigma^*$ generated by G. (i.e. $\mathcal{L}(G) = \{s \in \Sigma^* \mid \delta(s, x_o)!\}$. Intuitively, this language corresponds to the uncontrolled behavior of the DES including the unexpected behaviors.

Given $s, s' \in \Sigma^*$, we say that $s' \leq s$ whenever s' is a prefix of s (i.e. it exists $t \in \Sigma^*$ s.t. $s = s't$). We denote by $\overline{L}$ the prefix-closure of a language $L \subseteq \Sigma^*$ ($\overline{L} = \{s \in \Sigma^* \mid \exists s' \in L, s \leq s'\}$). Note that $\mathcal{L}(G)$, as defined above, is prefix-closed (i.e. $\mathcal{L}(G) = \overline{\mathcal{L}(G)}$).

For $L \subseteq \Sigma^*$ and $\Sigma' \subseteq \Sigma$, we use $L(s, \Sigma')$ to denote the set of suffixes of L after s that belongs to Σ'^*, i.e. $L(s, \Sigma') = \{t \in \Sigma'^* \mid st \in L\}$ and we note $L(s)$ for $L(s, \Sigma)$.

Given a plant to be controlled, some of its events in Σ are said to be uncontrollable (Σ_{uc}), i.e., their occurrence cannot be prevented by a supervisor, while the others are controllable (Σ_c). First, we recall the definition of a *controllable language* [8].

Definition 2 *Let G be an FSM modeling the plant and $K \subseteq \mathcal{L}(G)$ the prefix-closed specification. Then K is controllable with respect to Σ_{uc} and G (or $\mathcal{L}(G)$) if*

$$K\Sigma_{uc} \cap \mathcal{L}(G) \subseteq K \tag{1}$$

The behavior restriction of G is achieved by means of a a feedback control (named *Supervisor*). Formally, a supervisor is given by a function $\mathcal{S} : \mathcal{L}(G) \rightarrow \{\gamma \in 2^{\Sigma}; \Sigma_{uc} \subseteq \gamma\}$, delivering the set of actions that are allowed in G by the control after a trajectory $s \in \mathcal{L}(G)$. Write $\mathcal{S}/G$ for the closed loop system, consisting of the initial plant G controlled by the supervisor $\mathcal{S}$. The closed-loop system $\mathcal{S}/G$ is a Discrete Event System that can be characterized by the language $\mathcal{L}(\mathcal{S}/G)$ which is recursively defined as follows:

1. $\epsilon \in \mathcal{L}(\mathcal{S}/G)$
2. $[(s \in \mathcal{L}(\mathcal{S}/G))\ and\ (s\sigma \in \mathcal{L}(G)\ \wedge\ \sigma \in \mathcal{S}(s))] \Rightarrow s\sigma \in \mathcal{L}(\mathcal{S}/G)$

In general, given an plant G and a (prefix-closed) specification $K \subseteq \mathcal{L}(G)$, K is not controllable w.r.t. $\mathcal{L}(G)$ and Σ_{uc}, which means that it is necessary to restrict the behavior of K in order to obtain a sublanguage of K that is controllable with respect to both $\mathcal{L}(G)$ and Σ_{uc}.

Supervisory Control Problem ([8]) *Given a plant modeled by an FSM G and an expected language $K \subseteq \mathcal{L}(G)$, the problem is to build a supervisor S such that $\mathcal{L}(\mathcal{S}/G) \subseteq K$ is controllable, and for any other supervisor $\mathcal{S}'$ s.t. $\mathcal{L}(\mathcal{S}'/G) \subseteq K \Rightarrow \mathcal{L}(\mathcal{S}'/G) \subseteq \mathcal{L}(\mathcal{S}/G)$.*

In the sequel, we will be more interested in the computation of $\mathcal{S}/G$ rather than in the computation of the supervisor $\mathcal{S}$ itself, since one can easily extract $\mathcal{S}$ from $\mathcal{S}/G$. The solution of this problem is classically called the supremal controllable sub-language of K w.r.t. Σ_{uc} and $\mathcal{L}(G) = L$ (see [8]) and is denoted $\mathcal{S}_K/G$ or $K^{\uparrow L,c}$ or $SupCont(K, L, \Sigma_{uc})$ in the remainder of this paper. From a computational point of view, when K is prefix-closed, it can be shown that

$$K^{\uparrow L,c} \quad = K \setminus [(L \setminus K)/\Sigma_{uc}^*]\Sigma^* \tag{2}$$

with $L_1/L_2 = \{s \in \Sigma^* \mid \exists t \in L_2,\ st \in L_1\}$ for L_1 and L_2 languages over Σ^*.

In some situations, it is also of interest to compute $K^{\downarrow L,c}$ the *infimal prefix-closed and controllable superlanguage of* K w.r.t. $\mathcal{L}(G)$ and Σ_{uc}, which basically corresponds to the smallest prefix-closed language that contains K and that is controllable w.r.t. Σ_{uc} and $\mathcal{L}(G) = L$. It can be shown (see e.g. [2]) that

$$K^{\downarrow L,c} \quad = K\Sigma_{uc}^* \cap L \tag{3}$$

2.2 Concurrent DES.

In this paper, our aim is to control a plant composed of several components, sharing common events. To do so, let us consider a plant G modeled as a collection of FSM $G_i = \langle \Sigma_i, \mathcal{X}_i, x_{oi}, \delta_i \rangle$. The global behavior of the plant is given by $G = G_1 \parallel \cdots \parallel G_n$, where the operation $\parallel$ is the classical *parallel composition* (i.e. $G_1 \parallel G_2$ represents the concurrent behavior of G_1 and G_2 with synchronization on the shared events). It is defined by:

Definition 3 *Let $G_i = (\Sigma_i, \mathcal{X}_i, x_{o_i}, \delta_i)$, $i = 1, 2$ be two FSM. The* parallel composition $G_1 \parallel G_2$ *of G_1 and G_2 is the FSM $G = (\Sigma, \mathcal{X}, x_o, \delta)$ where $\Sigma = \Sigma_1 \cup \Sigma_2$, $\mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2$, $x_o = \langle x_{o_1}, x_{o_2} \rangle$, and δ is defined by: for all $x = \langle x_1, x_2 \rangle \in \mathcal{X}$ and $\sigma \in \Sigma$*

$$\delta(\sigma, \langle x_1, x_2 \rangle) = \begin{cases} \langle \delta_1(\sigma, x_1), x_2 \rangle & \text{if } \sigma \in \delta_1(x_1) \setminus \Sigma_2 \\ \langle x_1, \delta_2(\sigma, x_2) \rangle & \text{if } \sigma \in \delta_2(x_2) \setminus \Sigma_1 \\ \langle \delta_1(\sigma, x_1), \delta_2(\sigma, x_2) \rangle & \text{if } \sigma \in \delta_1(x_1) \cap \delta_2(x_2) \\ \text{Undefined} & \text{otherwise} \end{cases}$$

Now, given the set of FSM $(G_i)_{i \leq n}$ modeling G, we denote by Σ_s the set of shared events of G, i.e

$$\Sigma_s = \{\sigma \in \Sigma \mid \exists i \neq j,\ \sigma \in \Sigma_i \cap \Sigma_j\}. \tag{4}$$

Let $\Sigma' \subseteq \Sigma$, then $P_{\Sigma'} : \Sigma^* \to \Sigma'^*$ is the natural projection from Σ^* to Σ'^* that erases in a sequence of Σ^* all the events that do not belong to Σ'. Formally,

$$\begin{cases} P_{\Sigma'}(\epsilon) & = \epsilon \\ P_{\Sigma'}(\sigma) & = \begin{cases} \epsilon & \text{if } \sigma \notin \Sigma_i \\ \sigma & \text{if } \sigma \in \Sigma_i \end{cases} \\ P_{\Sigma'}(s\sigma) & = P_{\Sigma'}(s) P_{\Sigma'}(\sigma) \text{ for } s \in \Sigma^* \text{ and } \sigma \in \Sigma \end{cases} \tag{5}$$

This definition is easily extended to the projection of regular languages as follows:

$$P_{\Sigma'}(L) = \{s' \in \Sigma'^* \mid \exists s \in L,\ s' = P_{\Sigma'}(s)\}$$

Given $L \subseteq \Sigma'^* \subseteq \Sigma^*$, the inverse projection is defined by $P_{\Sigma'}^{-1}(L) = \{s \in \Sigma^* \mid P_{\Sigma'}(s) \in L\}$. From an implementation point of view, if H denotes the FSM such that $\mathcal{L}(H) = L$, then the FSM modeling the inverse projection of L, noted H^{-1}, can be obtained from H by simply adding self-loops labeled by events in $\Sigma \setminus \Sigma'$ to each state of H.

Given a concurrent discrete event system $G = G_1 \parallel \cdots \parallel G_n$, with $\mathcal{L}(G_i) \subseteq \Sigma_i^*$, we simply denote by P_i the projection from Σ^* to Σ_i^* and by P_i^{-1} the inverse projection from Σ_i^* to Σ^*. Based on these operations, the language resulting from the parallel composition of FSM is characterized by:

$$\mathcal{L}(G) = \mathcal{L}(G_1 \parallel \cdots \parallel G_n) = \mathcal{L}(G_1) \parallel \cdots \parallel \mathcal{L}(G_n) = P_1^{-1}[\mathcal{L}(G_1)] \cap \cdots \cap P_n^{-1}[\mathcal{L}(G_n)] \tag{6}$$

Note that we use the same $\parallel$ notation for the parallel composition of languages. The following technical lemmas will be useful in the sequel

Lemma 1 *Let $L \subseteq \Sigma'^*$ and $\Sigma' \subseteq \Sigma$, let $s \in L$ and $s' \in \Sigma^*$, then $ss' \in P_{\Sigma'}^{-1}(L) \Longrightarrow sP_{\Sigma'}(s') \in P_{\Sigma'}^{-1}(L)$*

Proof:
$$\begin{aligned} ss' \in P_{\Sigma'}^{-1}(L) & \Rightarrow P_{\Sigma'}(ss') \in L \Rightarrow P_{\Sigma'}(s)P_{\Sigma'}(s') \in L \\ & \Rightarrow P_{\Sigma'}(s)P_{\Sigma'}(P_{\Sigma'}(s')) \in L \\ & \Rightarrow P_{\Sigma'}(sP_{\Sigma'}(s')) \in L \\ & \Rightarrow sP_{\Sigma'}(s') \in P_{\Sigma'}^{-1}(L) \end{aligned}$$

Lemma 2 ((3.1) of [11]) *Let $G = G_1 \parallel \cdots \parallel G_n$, $s \in \mathcal{L}(G)$, $i \in \{1, \ldots, n\}$ and $\sigma \in \Sigma_i \setminus \Sigma_s$. Then*

$$s\sigma \in \mathcal{L}(G) \Longleftrightarrow s\sigma \in P_i^{-1}(\mathcal{L}(G_i))$$

2.3 Control Problem formulation & Related Works

Let $G = G_1 \parallel \cdots \parallel G_n$ be the plant to be controlled and $L_i = \mathcal{L}(G_i)$ be the language generated by the component G_i for $i \leq n$.

Event status. The alphabet of G_i is partionned into the controllable event set $\Sigma_{i,c}$ and the uncontrollable event set $\Sigma_{i,uc}$, i.e. $\Sigma_i = \Sigma_{i,uc} \dot{\cup} \Sigma_{i,c}$. The alphabet of the global plant G is given by:

$$\Sigma = \bigcup_i \Sigma_i, \Sigma_c = \bigcup_i \Sigma_{i,c}, \text{ and } \Sigma_{uc} = \Sigma \setminus \Sigma_c.$$

Moreover, we assume that the following relation holds between the control status of shared events:

$$\forall i, j,\ \Sigma_{i,uc} \cap \Sigma_{j,c} = \emptyset \tag{7}$$

which simply means that the components that share an event agree on the control status of this event. In other words, the control status of the events is the same for all the components. Under this hypothesis, we have that

$$\Sigma_{uc} = \bigcup_i \Sigma_{i,uc}$$

The control problem. Let $K \subseteq \Sigma^*$ be the expected behavior. The problem, we are interested in, is the Basic Supervisory Control Problem, i.e . the problem is to compute the supremal controllable sublanguage $(K \cap \mathcal{L}(G))^{\uparrow c}$ of $K \cap \mathcal{L}(G)$ w.r.t. $\mathcal{L}(G)$. However, knowing that the synthesis algorithms are polynomial in the number of states of G and that the size of the state space of G is exponential in the number of components of G, it is important to design algorithms that perform the controller synthesis phase by taking advantage of the structure of G without building it. Hence, the actual problem is to compute $(K \cap \mathcal{L}(G))^{\uparrow c}$ without computing neither $\mathcal{L}(G)$ nor $K \cap \mathcal{L}(G)$.

2.3.1 A Decentralized approach

The works of [11] is closely related to the decentralized theory. The authors consider the control of Concurrent Discrete Event Systems $G_1 \parallel \cdots \parallel G_n$. Given a language-based specification K, they provide some solutions allowing to compute local modular supervisors $\mathcal{S}_i$ on G_i (based on a notion of separable specification (See Definition 4)) and to operate the individually controlled system $\mathcal{S}_i/G_i$ concurrently in such a way that the controlled behavior corresponds to the supremal controllable sublanguage of $K \cap \mathcal{L}(G)$ w.r.t. $\mathcal{L}(G)$.

Definition 4 *$\mathcal{L} \subseteq \Sigma^*$ is said to be separable w.r.t. $\{\Sigma_i\}_{i \leq n}$ with $\cup_{i \leq n} \Sigma_i = \Sigma$, whenever there exists a set of languages $\{\mathcal{L}_i\}_{i \leq n}$ (called generating set), s.t. $\mathcal{L}_i \subseteq \Sigma_i^*$ and $\mathcal{L} = \mathcal{L}_1 \parallel \cdots \parallel \mathcal{L}_n$*[1]*.*

Based on this definition, Wilnner and Heymann shown that

Theorem 1 *Let $G = G_1 \parallel \cdots \parallel G_n$, with $\mathcal{L}(G_i) \subseteq \Sigma_i^*$. and K the expected specification. If $\Sigma_s \subseteq \Sigma_c$ and K is separable w.r.t. $\{\Sigma_i\}_{i \leq n}$, then*

$$\|_{i \leq n}\ SupC(P_i(K) \cap \mathcal{L}(G_i), \mathcal{L}(G_i), \Sigma_{i,uc}) = SupC(K \cap \mathcal{L}(G), \mathcal{L}(G), \Sigma_{uc})$$

Hence, given a Concurrent DES G and a separable specification K, Theorem 1 shows that there exists a set of supervisors $\mathcal{S}_i$ acting upon G_i, such that $\|_{i \leq n}\ \mathcal{L}(S_i/G_i) = (K \cap \mathcal{L}(G))^{\uparrow c}$. The supervisors architecture is given in Figure 1.

If K is separable w.r.t. $\{\Sigma_i\}_{i \leq n}$ (which can be checked in $\mathcal{O}(m^{n+1})$, where m is the size of the FSM that generates K), then to synthesize the local supervisors it is needed to compute the projection of K over

[1] with $\mathcal{L}_1 \parallel \mathcal{L}_1 = P_1^{-1}(\mathcal{L}_1) \cap P_2^{-1}(\mathcal{L}_2)$.

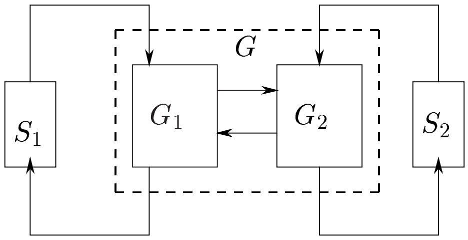


Figure 1: A Decentralized architecture for Concurrent DES

Σ_i. In the worst case, the size of the FSM that generate $P_i(K)$ is in $\mathcal{O}(2^m)$. Hence, solving the supervisory control problem will require $\mathcal{O}(n.2^m.N)$ space where N is the size of each component. Solving the SCP by first computing would have required $\mathcal{O}(N^n.m)$ space.
In [1], the authors adopt a similar approach. They do not require the specification K to be separable w.r.t. $\{\Sigma_i\}_{i\leq n}$. K is in fact decomposed as $K = [\|_{i\leq n} P_i(K)] \cap K_{cp}$, where $K_{cp} \subseteq \Sigma^*$ is called a *compensator*. Further it is shown that if $\Sigma_s \subseteq \Sigma_c$ and if $K_{cp}\Sigma_u \subseteq K_{cp}$, then

$$SupCont(K \cap \mathcal{L}(G), \mathcal{L}(G), \Sigma_{uc}) = \{\|_{i\leq n} SupC(P_i(K), \mathcal{L}(G_i), \Sigma_{uc})\} \cap K_{cp}$$

Hence, all the problem, it to find a compensator K_{cp} that respect the condition $K_{cp}\Sigma_u \subseteq K_{cp}$.

2.3.2 Our approach

Our approach is different and is more related to the modular approach of [12]. Indeed, the plant G can be described by the following parallel composition of FSM $G =\|_{i\leq n} G_i^{-1}$, where G_i^{-1} is the FSM such that $\mathcal{L}(G_i^{-1}) = P_i^{-1}(\mathcal{L}(G_i))$. In fact, each G_i^{-1} can be seen as an approximation of the plant G to be controlled.

Compared to [11], we adopt a dual approach. Instead of controlling w.r.t. each component G_i (i.e. $\mathcal{L}(G_i)$) to enforce $P_i(K)$, we have chosen to control the approximations $\mathcal{L}(G_i^{-1})$ of the plant in order to enforce K. However, it is not sufficient to compute a supervisor $\mathcal{S}_i$ acting upon G_i^{-1} that restricts the behavior $\mathcal{L}(G_i^{-1})$ to the supremal controllable sublanguage of $K \cap \mathcal{L}(G_i^{-1})$ and to operate the controlled systems $\mathcal{S}/G_i^{-1}$ concurrently to obtain the supremal controllable sublanguage of $K \cap \mathcal{L}(G)$ (the result may be not supremal). So the idea of our method is to refine the notion of controllability.

The property that we ensure on each G_i^{-1} according to K is called *the partial controllability condition* and is defined in Section 3. As in the case of the controllability concept, it will be shown that there exists a supremal partially controllable sublanguage of $K \cap \mathcal{L}(G_i^{-1})$ w.r.t. K and G_i^{-1}, called $K_i^{\uparrow pc}$. It is then shown that $\cap_{i\leq n} K_i^{\uparrow pc} \subseteq (K \cap \mathcal{L}(G))^{\uparrow c}$ (Theorem 2) and that under some conditions on K the equality holds (Theorem 3 and 4). A comparison with the results of [11] is then done. The control architecture is summarized in Figure 5 of Section 4.3.

3 Partial Controllability Property

In this section, we introduce a new concept of controllability, named *Partial Controllability*, that will serve as the bases of the modular computation of supervisors for Concurrent discrete event systems.

3.1 Definition and useful properties

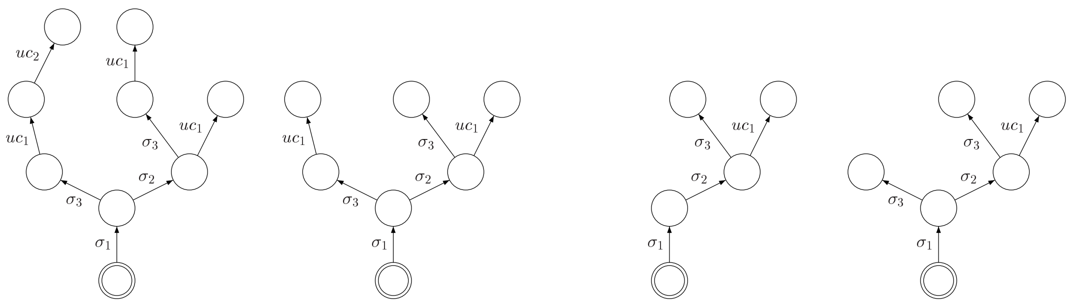
Definition 5 *Let $M \subseteq L \subseteq \Sigma^*$ be prefix-closed languages. Let $\Sigma'_{uc} \subseteq \Sigma_{uc} \subseteq \Sigma$ be two sub-alphabets of Σ. Let $M' \subseteq M$, then M' is partially controllable with respect to Σ'_{uc}, Σ_{uc}, M and L if*

(i) M' is controllable w.r.t Σ'_{uc} and L.

(ii) M' is controllable w.r.t Σ_{uc} and M.

We now intuitively explain this definition via the example 1

Example 1 *To illustrate this definition, let us consider the languages L, $M = (M_1)$, M_2 and M_3 described in Figure 2. Assume that $\Sigma_{uc} = \{uc_1, uc_2\}$ and $\Sigma'_{uc} = \{uc_2\}$.*



(a) L (b) M $(= M_1)$ (c) M_2 (d) M_3

Figure 2: The behaviors of L, M, M_2, M_3.

- *$M_1 = M$ is not partially controllable w.r.t. Σ'_{uc}, Σ_{uc}, M and L, since M is not controllable w.r.t Σ'_{uc} and L. Indeed, $\sigma_1\sigma_3 uc_1 \in M_1$ and $\sigma_1\sigma_3 uc_1 uc_2 \in L$ with $uc_2 \in \Sigma'_{uc}$, but $\sigma_1\sigma_3 uc_1 uc_2 \notin M_1$. However, $M_1 = M$ is trivially controllable w.r.t Σ_{uc} and M.*
- *A contrario, M_2 is partially controllable w.r.t. Σ'_{uc}, Σ_{uc}, M and L. Indeed, on one hand, M_2 is controllable w.r.t. Σ'_{uc}, and L (note that, since $uc_1 \notin \Sigma'_{uc}$, hence $\sigma_1\sigma_2\sigma_3 uc_1$ is not required to belong to M_2). On the other hand M_2 is controllable w.r.t. Σ_{uc}, and M. However, note that M_2 is not controllable w.r.t. Σ_{uc}, and L.*
- *Finally, M_3 is controllable w.r.t. Σ'_{uc}, and L. But M_3 is not partially controllable w.r.t. Σ'_{uc}, Σ_{uc}, M and L. Indeed, M_3 is not controllable w.r.t. Σ_{uc}, and M since $\sigma_1\sigma_3 \in M_3$, $\sigma_1\sigma_3 uc_1 \in M$ with $uc_1 \in \Sigma_{uc}$, but $\sigma_1\sigma_3 uc_1 \notin M_3$.* $\diamond$

In general, M is not partially controllable with respect to Σ'_{uc}, Σ_{uc}, M and L (e.g. if M is not controllable w.r.t. Σ'_{uc} and L (see Example 1)). However, it can be shown that there exists a supremal sub-language of M that has this property.

Proposition 1 *Let $M \subseteq L \subseteq \Sigma^*$ be prefix-closed languages, $\Sigma'_{uc} \subseteq \Sigma_{uc}$. There exists a unique supremal language, denoted by $M^{\uparrow pc}$, which is partially controllable w.r.t Σ'_{uc}, Σ_{uc},M and L. Moreover*

$$M^{\uparrow pc} = \overline{M^{\uparrow pc}} = \mathit{SupCont}(\mathit{SupCont}(M, \Sigma'_{uc}, L), \Sigma_{uc}, M) \tag{8}$$

Proof : First of all, the set of partially controllable languages w.r.t Σ'_{uc}, Σ_{uc}, M and L is not empty, since $\emptyset$ is partially controllable w.r.t Σ'_{uc}, Σ_{uc}, M and L. Now let $M' \subseteq M$ be a partially controllable language w.r.t Σ'_{uc}, Σ_{uc}, M and L. Since $M' \subseteq M$ is controllable w.r.t Σ'_{uc} and L, $M' \subseteq \mathrm{SupCont}(M, \Sigma'_{uc}, L)$, and by monotony of the $\mathrm{SupCont}(, \Sigma_{uc}, M)$ operator, we have that

$$\mathrm{SupCont}(M', \Sigma_{uc}, M) \subseteq \mathrm{SupCont}(\mathrm{SupCont}(M, \Sigma'_{uc}, L), \Sigma_{uc}, M)$$

Now, $M' \subseteq M$ is controllable w.r.t Σ_{uc} and M. This entails that $M' = \mathrm{SupCont}(M', \Sigma_{uc}, M)$, and finally

$$M' \subseteq \mathrm{SupCont}(\mathrm{SupCont}(M, \Sigma'_{uc}, L), \Sigma_{uc}, M)$$

Let us now show that $\mathrm{SupCont}(\mathrm{SupCont}(M, \Sigma'_{uc}, L), \Sigma_{uc}, M)$ is partially controllable w.r.t Σ'_{uc}, Σ_{uc}, M and L.

(i) let $s \in \text{SupCont}(\text{SupCont}(M, \Sigma'_{uc}, L), \Sigma_{uc}, M)$ and $\sigma \in \Sigma'_{uc}$ such that $s\sigma \in L$. As

$$\text{SupCont}(\text{SupCont}(M, \Sigma'_{uc}, L), \Sigma_{uc}, M) \subseteq \text{SupCont}(M, \Sigma'_{uc}, L)$$

we obtain $s\sigma \in \text{SupCont}(M, \Sigma'_{uc}, L)\Sigma'_{uc} \cap L$. Hence, by definition of controlability, $s\sigma \in \text{SupCont}(M, \Sigma'_{uc}, L)$ and as $\text{SupCont}(M, \Sigma'_{uc}, L) \subseteq M$, we also have that $s\sigma \in M$.

Moreover, since $\Sigma'_{uc} \subseteq \Sigma_{uc}$, we obtain that $s\sigma \in \text{SupCont}(\text{SupCont}(M, \Sigma'_{uc}, L), \Sigma_{uc}, M)\Sigma_{uc} \cap M$. Hence, by definition of controllability again,

$$s\sigma \in \text{SupCont}(\text{SupCont}(M, \Sigma'_{uc}, L), \Sigma_{uc}, M)$$

Thus $\text{SupCont}(\text{SupCont}(M, \Sigma'_{uc}, L), \Sigma_{uc}, M)$ is controllable w.r.t. Σ'_{uc} and L.

(ii) By definition, $\text{SupCont}(\text{SupCont}(M, \Sigma'_{uc}, L), \Sigma_{uc}, M)$ is controllable w.r.t Σ_{uc} et M.

Finally $\text{SupCont}(\text{SupCont}(M, \Sigma'_{uc}, L), \Sigma_{uc}, M)$ is partially controllable w.r.t Σ'_{uc}, Σ_{uc}, M et L. Moreover, it contains any partially controllable language w.r.t Σ'_{uc}, Σ_{uc}, M et L. Therefore, it is the supremal (and we denote it by $M^{\uparrow pc}$. The fact that $M^{\uparrow pc}$ is prefix-closed is trivial since the SupC() operator preserves the prefix closure. $\diamond$

Coming back to Example 1, it is easy to check that the supremal partially controllable sub-language of M w.r.t. to Σ'_{uc}, Σ_{uc}, and L is reduced to the M_2 language.
Theorem 1 offers a practical way to compute the supremal partially controllable sub-language of M w.r.t. to Σ'_{uc}, Σ_{uc}, and L. It can then be shown that the computation of the supremal partially controllable sub-languages of M w.r.t. to Σ'_{uc}, Σ_{uc}, and L is in $\mathcal{O}(N_M^2 N_L)$, where N_M is the size of the automata encoding M and N_L the one of L.

4 Control of Concurrent Discrete Event Systems

Given a Concurrent DES, $G = G_1 \parallel \cdots \parallel G_n$, and a control objective K, we want to compute a controllable sub-language of $K \cap \mathcal{L}(G)$ w.r.t. $\mathcal{L}(G)$ and Σ_{uc}, without having to build G itself.

4.1 Modular Computation of a controllable sub-language of K w.r.t. $\mathcal{L}(G)$

Based on the concept of partial controllability applied on K and on the approximations of the plant $P_i^{-1}(\mathcal{L}(G_i))$ derived from each of its components, the next theorem provides a modular way to compute a sub-language of K that is controllable with respect to the plant.

Theorem 2 *Let $G = G_1 \parallel \cdots \parallel G_n$, with G_i acting upon $\Sigma_i = \Sigma_{i,uc} \cup \Sigma_{i,c}$ and $L_i = \mathcal{L}(G_i)$. Let $K \subseteq \Sigma^*$ be a prefix-closed language modeling the expected behavior. For $i \leq n$, we note*

- *$K_i = K \cap P_i^{-1}(\mathcal{L}(G_i))$, and*
- *$K_i^{\uparrow pc}$ the supremal sublanguage of K_i partially controllable with respect to $\Sigma_{i,uc}$, Σ_{uc}, K_i and $P_i^{-1}(\mathcal{L}(G_i))$.*

Then, $K_1^{\uparrow pc} \cap \cdots \cap K_n^{\uparrow pc}$ is controllable with respect to Σ_{uc} and $\mathcal{L}(G)$.

Proof : First, as K, $\mathcal{L}(G_i)$ are prefix-closed, languages $P_i^{-1}(\mathcal{L}(G_i))$ for $i \leq n$ are prefix-closed, and therefore K_i and $K_i^{\uparrow pc}$ are also prefix-closed. Now, according to Definition 2, we have to show that

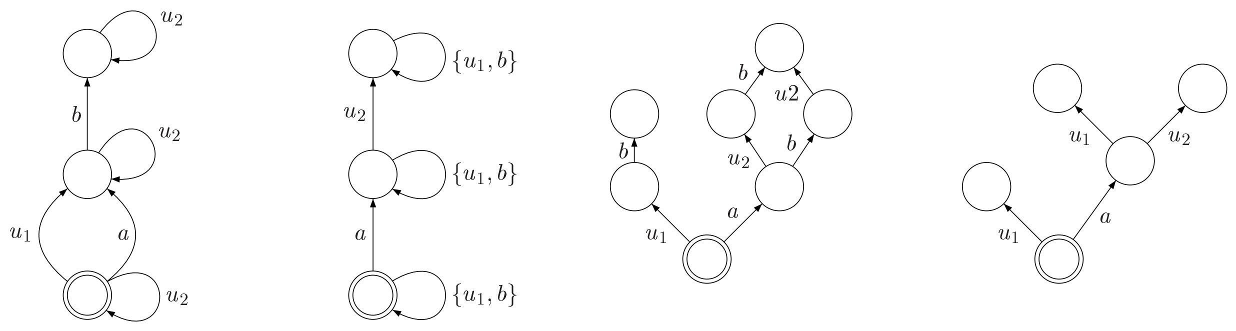
$$(\bigcap_{i \leq n} K_i^{\uparrow pc})\Sigma_{uc} \cap \mathcal{L}(G) \subseteq \bigcap_{i \leq n} K_i^{\uparrow pc}$$

Let $s \in \bigcap_{i \leq n} K_i^{\uparrow pc}$ and $\sigma \in \Sigma_{uc}$ be such that $s.\sigma \in \mathcal{L}(G)$. We thus have to show that $s\sigma \in \bigcap_{i \leq n} K_i^{\uparrow pc}$. Without lost of generality we can assume that $\sigma \in \Sigma_{1,uc}$.

Since $s\sigma \in \mathcal{L}(G)$, $s\sigma \in P_1^{-1}(\mathcal{L}(G_1))$. Hence we have that $s\sigma \in K_1^{\uparrow pc}\Sigma_{1,uc} \cap P_1^{-1}(\mathcal{L}(G_1))$. Moreover, according to definition 5, $K_1^{\uparrow pc}$ is controllable w.r.t. $\Sigma_{1,uc}$ and $P_1^{-1}(\mathcal{L}(G_1))$, from which we can conclude that $s\sigma \in K_1^{\uparrow pc}$.

As $s\sigma \in K_1^{\uparrow pc}$, we have that $s\sigma \in K$. And as $s\sigma \in \mathcal{L}(G)$, then $\forall i \leq n,\ s\sigma \in P_i^{-1}(\mathcal{L}(G_i))$, which entails that $\forall i \leq n$, $s\sigma \in K_i = K \cap P_i^{-1}(\mathcal{L}(G_i))$. Hence, $\forall i \leq n$, $s\sigma \in {K_i}^{\uparrow pc}\Sigma_{uc} \cap K_i$. Now, according to definition 5, $\forall i \leq n,\ K_i^{\uparrow pc}$ is controllable w.r.t. Σ_{uc} and K_i. Hence $\forall i \leq n,\ s\sigma \in K_i^{\uparrow pc}$, which entails that $s\sigma \in \bigcap_{i \leq n} K_i^{\uparrow pc}$. ⋄

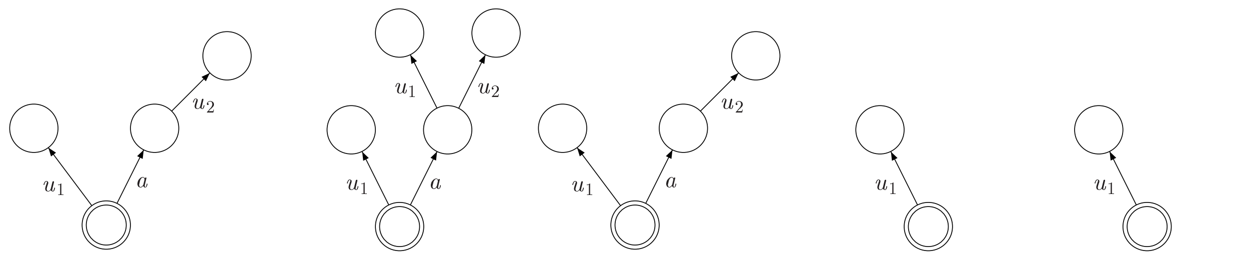
Example 2 *Let $\mathcal{L}(G) = \mathcal{L}(G_1) \parallel \mathcal{L}(G_2)$, with $\mathcal{L}(G_1) = \overline{\{(a + u_1)b\}}$ and $\mathcal{L}(G_2) = \{a, au_2\}$. We have $\Sigma_1 = \{a, u_1, b\}$, $\Sigma_2 = \{a, u_2\}$. Figure 3(a) and 3(b) represent the FSM generating $P_1^{-1}(\mathcal{L}(G_1))$ and $P_2^{-1}(\mathcal{L}(G_2))$, whereas Figure 3(c) is the FSM G (note that the FSM G_1 and G_2 can be easily obtained from Figure 3(a) and 3(b) by removing the self-loops). Finally, in this example, Σ_s is reduced to the singleton $\{a\}$. Assume the control objective is given by the language K as described in Figure 3(d).*



(a) $P_1^{-1}(\mathcal{L}(G_1))$ (b) $P_2^{-1}(\mathcal{L}(G_2))$ (c) $\mathcal{L}(G)$ (d) K

Figure 3: The behaviors of the system.

Following Theorem 2, we first compute the languages $K_i = K \cap P_i^{-1}(\mathcal{L}(G_i))$, $i = 1, 2$. The FSM generating K_1 and K_2 are represented in Figures 4(a) and 4(b).



(a) K_1 (b) K_2 (c) $K_1^{\uparrow pc}$ (d) $K_2^{\uparrow pc}$ (e) $K_1^{\uparrow pc} \cap K_2^{\uparrow pc}$

Figure 4: The control objective & the derived local ones

Now based on (8), we compute $K_1^{\uparrow pc}$ (resp $K_2^{\uparrow pc}$), the supremal language of K_1 (resp. K_2) that is partially controllable w.r.t. $P_1^{-1}(\mathcal{L}(G_1)$, $\Sigma_{uc} = \{u_1, u_2\}$ and $\Sigma_{1,uc} = \{u_1\}$ (resp $\Sigma_{2,uc} = \{u_2\}$ and $P_2^{-1}(\mathcal{L}(G_2))$. The intersection of these two languages leads to the language $K_1^{\uparrow pc} \cap K_2^{\uparrow pc}$ (Figure 4(e)), that is obviously controllable w.r.t. $\mathcal{L}(G)$ and Σ_{uc}. ⋄

4.2 Computation of $(K \cap \mathcal{L}(G))^{\uparrow c}$

According to Theorem 2, we have that $\bigcap_{i \leq n} K_i^{\uparrow pc} \subseteq (K \cap \mathcal{L}(G))^{\uparrow c}$, where $(K \cap \mathcal{L}(G))^{\uparrow c}$ corresponds to the supremal sublanguage of $K \cap \mathcal{L}(G)$ that is controllable w.r.t. Σ_{uc} and $\mathcal{L}(G)$. But, the equality does not hold in general (See Example 2, in which the supremal controllable sublanguage of $K \cap \mathcal{L}(G)$ is given by $(K \cap \mathcal{L}(G))^{\uparrow c} = \{a, u_1, au_2\}$). In this section, we present some conditions under which Theorem 2 gives access to the supremal solution.
Let us first introduce lemma 3. This lemma shows that whenever the shared events Σ_s are controllable, then $(K \cap \mathcal{L}(G))^{\uparrow c}$ verifies a part of the partial controllability condition.

Lemma 3 *Let $G = G_1 \parallel \cdots \parallel G_n$ be the plant and $K \subseteq \Sigma^*$ a prefix-closed language modeling the expected behavior, then if $\Sigma_s \subseteq \Sigma_c$, then $\forall i \leq n$, $(K \cap \mathcal{L}(G))^{\uparrow c}$ is controllable w.r.t $\Sigma_{i,uc}$, $P_i^{-1}(\mathcal{L}(G_i))$.*

Proof : Let $i \leq n$. Le us consider $s \in (K \cap \mathcal{L}(G))^{\uparrow c}$ and $\sigma \in \Sigma_{i,uc}$ such that $s\sigma \in P_i^{-1}(\mathcal{L}(G_i))$. We have to show that $s\sigma \in (K \cap \mathcal{L}(G))^{\uparrow c}$. Since $\sigma \in \Sigma_{i,uc}$ and $\Sigma_s \subseteq \Sigma_c$, we have $\sigma \in \Sigma_i \setminus \Sigma_s$. Moreover, $s \in P_i^{-1}(\mathcal{L}(G_i))$, $s \in \mathcal{L}(G)$ and $s\sigma \in P_i^{-1}(\mathcal{L}(G_i))$, which entails that $s\sigma \in \mathcal{L}(G)$ (Lemma 2). Now, as $\Sigma_{i,uc} \subseteq \Sigma_{uc}$, we have that $s\sigma \in (K \cap \mathcal{L}(G))^{\uparrow c}.\Sigma_{uc} \cap \mathcal{L}(G)$. Since $(K \cap \mathcal{L}(G))^{\uparrow c}$ is controllable w.r.t Σ_{uc} and $\mathcal{L}(G)$, this entails that $s\sigma \in (K \cap \mathcal{L}(G))^{\uparrow c}$. $\diamond$

Theorem 3 *If $\Sigma_s \subseteq \Sigma_c$ and $K \subseteq \mathcal{L}(G)$, then with the notations of Theorem 2, $\bigcap_{i \leq n} K_i^{\uparrow pc} = K^{\uparrow c}$.*

Proof : From Theorem 2, $\bigcap_{i \leq n} K_i^{\uparrow pc} \subseteq K^{\uparrow c}$. It is then sufficient to show that $K^{\uparrow c} \subseteq \bigcap_{i \leq n} K_i^{\uparrow pc}$ or, equivalently that $\forall i \leq n$, $K^{\uparrow c} \subseteq K_i^{\uparrow pc}$. To do so, let us pick up a $i \leq n$, and let us show that $K^{\uparrow c}$ is partially controllable w.r.t. $\Sigma_{i,uc}$, Σ_{uc}, K_i and $P_i^{-1}(L_i)$.

(i) First, according to lemma 3, $K^{\uparrow c}$ is controllable w.r.t $\Sigma_{i,uc}$ and $P_i^{-1}(\mathcal{L}(G_i))$.

(ii) Let us now show that $K^{\uparrow c}$ is controllable w.r.t. Σ_{uc} and K_i. Let us consider $s \in K^{\uparrow c}$, $\sigma \in \Sigma_{uc}$ such that $s\sigma \in K_i$, we have to prove that $s\sigma \in K^{\uparrow c}$. Since $s\sigma \in K_i \subseteq \mathcal{L}(G)$, $s\sigma \in \mathcal{L}(G)$. We then have $s \in K^{\uparrow c}$, $\sigma \in \Sigma_{uc}$ and $s\sigma \in \mathcal{L}(G)$. Hence, because $K^{\uparrow c}$ is controllable w.r.t. Σ_{uc} and $\mathcal{L}(G)$, $s\sigma \in K^{\uparrow c}$.

This proves that $K^{\uparrow c}$ is partially controllable w.r.t. $\Sigma_{i,uc}$, Σ_{uc}, K_i, $P_i^{-1}(\mathcal{L}(G_i))$. Now, as $K_i^{\uparrow pc}$ is supremal and partially controllable w.r.t. $\Sigma_{i,uc}$, Σ_{uc}, K_i, $P_i^{-1}(\mathcal{L}(G_i)$, we can deduce that $\forall i \leq n$, $K^{\uparrow c} \subseteq K_i^{\uparrow pc}$. Finally, $K^{\uparrow c} \subseteq \bigcap_{i \leq n} K_i^{\uparrow pc}$. $\diamond$

This theorem states that whenever the language of the control objective is included in the one of the plant then our methodology gives access to the supremal controllable sub-language of K w.r.t. L and Σ_{uc}. As previously mentioned, the interest of this method is that it avoids the building of the entire plant; hence reducing the complexity of the supervisory synthesis phase. Indeed, if $G = G_1 \parallel \cdots \| G_n$ is such that $|\mathcal{X}_{G_i}| = N$ and H, with $|\mathcal{X}_H| = m$, is the FSM modeling the language specification K, then the FSM modeling the *partial specification* K_i are in $\mathcal{O}(N.m)$. According to Section 3, the complexity to compute the supremal partially controllable sublanguage of each K_i is in $\mathcal{O}(N^2.m)$. Finally, the overall complexity is in $\mathcal{O}(n.N^2.m)$. This has to be opposed to the space complexity $\mathcal{O}(N^n.m)$ of computing $(K \cap \mathcal{L}(G))^{\uparrow c}$ on G, seen as a unique FSM.

Remark 1 *To check that $K \subseteq \mathcal{L}(G)$ it is sufficient to check that $\forall 1 \leq i \leq n,\ K \subseteq P_i^{-1}(\mathcal{L}(G_i))$. Hence, it is not necessary to compute $\mathcal{L}(G)$.*

In some situations, modeling the expected behavior by a language included in the one of the plant may lead to a language that is too large to be efficiently represented. Moreover, requiring the inclusion of languages makes that the specification of K may be itself relatively difficult to identify insofar as the language L is not known. Theorem 4 gives another sufficient condition under which Theorem 2 gives access to the supremal solution. First, we need to introduce the notion of observable language.

Definition 6 *Let K and L be two prefix-closed languages over Σ and $\Sigma', \Sigma'' \subseteq \Sigma$. K is said to be observable w.r.t. $P_{\Sigma'}$, Σ'' and L if $\forall s, s' \in K$, $\forall \sigma \in \Sigma''$, if $P_{\Sigma'}(s') = P_{\Sigma'}(s)$ and $s'\sigma \in K$, and $s\sigma \in L$, then $s\sigma \in K$.*

It is shown in (e.g. [[2], Chap. 3.7] that the observability condition can be checked in $\mathcal{O}(m^2 N)$, where m and N are the number of states of the FSM that generate K and M.

Definition 7 *Let $K \subseteq \Sigma^*$ be a prefix-closed language and $G = G_1 \parallel \cdots \parallel G_n$ a concurrent system, then K is said to be G-observable if*

$$\forall i \in \{1, \ldots, n\},\ \forall s \in \Sigma^*,\ K_i(s, \Sigma_{uc}) \text{ is observable w.r.t. } P_i, \Sigma_{i,uc} \text{ and } P_i^{-1}(\mathcal{L}(G_i))(s, \Sigma_{uc}).$$

where $K_i = K \cap \mathcal{L}(G_i)$, $\forall i \leq n$.

Hence, the G-observability condition can also be checked in $\mathcal{O}(nm^3 N^4)$, where n is the number of components in G, N the state size of G_i and m the size of the FSM that generates K. Now, based on definition 7, we have that

Theorem 4 *Assume that $\Sigma_s \subseteq \Sigma_c$ and that K is G-observable, then $\bigcap_{1 \leq i \leq n} K_i^{\uparrow pc} = (K \cap \mathcal{L}(G))^{\uparrow c}$* $\diamond$

Proof : According to Theorem 2, $\bigcap_{i \leq n} K_i^{\uparrow pc} \subseteq (K \cap \mathcal{L}(G))^{\uparrow c}$ Thus, we have to show that

$$\forall i \leq n,\ (K \cap \mathcal{L}(G))^{\uparrow c} \subseteq K_i^{\uparrow pc}$$

To do so, let us consider $i \leq n$ and $((K \cap \mathcal{L}(G))^{\uparrow c})^{\downarrow K_i, c}$ (the infimal prefix-closed and controllable superlanguage of $(K \cap \mathcal{L}(G))^{\uparrow c}$ w.r.t. Σ_{uc} and K_i). We now prove that $((K \cap \mathcal{L}(G))^{\uparrow c})^{\downarrow K_i, c}$ is partially controllable w.r.t $\Sigma_{i,uc}$, Σ_{uc}, K_i and $P_i^{-1}(\mathcal{L}(G_i))$. Indeed, if $((K \cap \mathcal{L}(G))^{\uparrow c})^{\downarrow K_i, c}$ is partially controllable w.r.t $\Sigma_{i,uc}$, Σ_{uc}, K_i and $P_i^{-1}(\mathcal{L}(G_i))$, then we will have that $((K \cap \mathcal{L}(G))^{\uparrow c})^{\downarrow K_i, c} \subseteq K_i{}^{\uparrow pc}$. Moreover, as $(K \cap \mathcal{L}(G))^{\uparrow c} \subseteq ((K \cap \mathcal{L}(G))^{\uparrow c})^{\downarrow K_i, c}$, we will also have that $(K \cap \mathcal{L}(G))^{\uparrow c} \subseteq K_i{}^{\uparrow pc}$ and the proof will be done.

Let us now show that $((K \cap \mathcal{L}(G))^{\uparrow c})^{\downarrow K_i, c}$ is partially controllable w.r.t $\Sigma_{i,uc}$, Σ_{uc}, K_i and $P_i^{-1}(\mathcal{L}(G_i))$. According to Definition 5, we have to show that $((K \cap \mathcal{L}(G))^{\uparrow c})^{\downarrow K_i, c}$ is *(i)* controllable w.r.t. $\Sigma_{i,uc}$ and $P_i^{-1}(\mathcal{L}(G_i))$ and *(ii)* controllable w.r.t. Σ_{uc} and K_i. Item *(ii)* is obvious since by the definition of the infimal controllable language w.r.t. Σ_{uc} and K_i, $((K \cap \mathcal{L}(G))^{\uparrow c})^{\downarrow K_i, c}$ is controllable w.r.t. Σ_{uc} and K_i. Let us now prove the point *(i)*.

Let us consider $s \in ((K \cap \mathcal{L}(G))^{\uparrow c})^{\downarrow K_i, c}$ and $\sigma \in \Sigma_{i,uc}$ such that $s\sigma \in P_i^{-1}(\mathcal{L}(G_i))$. We have

$$s\sigma \in ((K \cap \mathcal{L}(G))^{\uparrow c})^{\downarrow K_i, c}.\Sigma_{i,uc} \cap P_i^{-1}(\mathcal{L}(G_i))$$

Moreover, according to (3),

$$((K \cap \mathcal{L}(G))^{\uparrow c})^{\downarrow K_i, c} = (K \cap \mathcal{L}(G))^{\uparrow c}.\Sigma_{uc}^* \cap K_i$$

So s is of the form $s = s't$ with $s' \in (K \cap \mathcal{L}(G))^{\uparrow c}$ and $t \in \Sigma_{uc}^*$. Now as $s\sigma = s't\sigma \in P_i^{-1}(\mathcal{L}(G_i))$, we also have that $s'P_i(t\sigma) \in P_i^{-1}(\mathcal{L}(G_i))$ (Lemma 1).

We now have that $s' \in (K \cap \mathcal{L}(G))^{\uparrow c}$, $P_i(t\sigma) \in \Sigma_{i,uc}^*$ and $s'P_i(t\sigma) \in P_i^{-1}(\mathcal{L}(G_i))$. We then have that $s'P_i(t\sigma) \in (K \cap \mathcal{L}(G))^{\uparrow c}$ as $(K \cap \mathcal{L}(G))^{\uparrow c}$ is controllable w.r.t. $\Sigma_{i,uc}$ and $P_i^{-1}(\mathcal{L}(G_i))$ (Lemma 3). Finally, as $s'P_i(t\sigma) \in (K \cap \mathcal{L}(G))^{\uparrow c} \subseteq K_i$, we obtain $P_i(t\sigma) = P_i(t)\sigma \in K_i(s', \Sigma_{uc})$.

Moreover, as by hypothesis $s\sigma = s't\sigma \in P_i^{-1}(\mathcal{L}(G_i))$, we also have that $t\sigma \in P_i^{-1}(\mathcal{L}(G_i))(s', \Sigma_{uc})$. By definition of the projection, $P_i(t\sigma) = P_i(P_i(t\sigma))$. Overall, we have that $P_i(t) \in K_i(s', \Sigma_{uc})$, $t \in K_i(s', \Sigma_{uc})$, with $P_i(t) = P_i(P_i(t))$, $\sigma \in \Sigma_{uc}$, $P_i(t\sigma) = P_i(t)\sigma \in K_i(s', \Sigma_{uc})$ and $t\sigma \in P_i^{-1}(\mathcal{L}(G_i))$. As $K_i(s', \Sigma_{uc})$ is observable w.r.t. $P_i, \Sigma_{i,uc}$ and $P_i^{-1}(\mathcal{L}(G_i))(s', \Sigma_{uc})$, we obtain that $t\sigma \in K_i(s', \Sigma_{uc})$. Hence $s\sigma\ (= s't\sigma) \in K_i$ and it entails that

$$s\sigma \in ((K \cap \mathcal{L}(G))^{\uparrow c})^{\downarrow K_i,c}.\Sigma_{uc} \cap K_i$$

As $((K \cap \mathcal{L}(G))^{\uparrow c})^{\downarrow K_i,c}$ is (by definition) controllable w.r.t. Σ_{uc} and K_i, $s\sigma \in ((K \cap \mathcal{L}(G))^{\uparrow c})^{\downarrow K_i,c}$. Thus $((K \cap \mathcal{L}(G))^{\uparrow c})^{\downarrow K_i,c}$ is controllable w.r.t. $\Sigma_{i,uc}$ and $P_i^{-1}(\mathcal{L}(G_i))$.

Finally, $((K \cap \mathcal{L}(G))^{\uparrow c})^{\downarrow K_i,c}$ is partially controllable w.r.t $\Sigma_{i,uc}$, Σ_{uc}, K_i and $P_i^{-1}(\mathcal{L}(G_i))$, which concludes the proof. $\diamond$

The condition of observability of Theorem 4 states that after each trace s admissible in K_i, if there exists two admissible suffixes of s of uncontrollable events, say s_1 and s_2, that have the same projection over the local one (i.e. $P_i(s_1) = P_i(s_2)$), then if one can be extended by an uncontrollable event in a trace of K_i and the other one in a trace of the abstracted specification G_i^{-1}, then the two extended traces have to belong to the specification. Finally, based on the previous results, the complexity of checking the condition of G-observability and computing $(K \cap \mathcal{L}(G))^{\uparrow}$ with our method is in $\mathcal{O}(n.m^2.N^2(m.N^2+1)$, which has to be opposed to $\mathcal{O}(N^n.m)$ (complexity with a monolithic approach).
The next proposition shows whenever K is separable then it respect the conditions of Theorem 4.

Proposition 2 *Let $G = G_1 \parallel \cdots \parallel G_n$ be the plant to be controlled s.t. $\mathcal{L}(G_i) \subseteq \Sigma_i^*$. Let $K \subseteq \Sigma^*$ be the expected specification. If K is separable w.r.t. $\{\Sigma_i\}_{1\leq i\leq n}$ then K is G-observable, i.e. $\forall i \leq n$, $\forall s \in K_i = K \cap P_i^{-1}(\mathcal{L}(G_i))$, $K_i(s, \Sigma_{uc})$ is observable w.r.t. P_i, $\Sigma_{i,uc}$ et $P_i^{-1}(\mathcal{L}(G_i))(s, \Sigma_{uc})$.*

Proof : As K is separable w.r.t. $\{\Sigma_i\}_{1\leq i\leq n}$, $K = \|_{i\leq n} \mathcal{E}_i$, with $\mathcal{E}_i = P_i(K)$ [11] and $\mathcal{E}_i \subseteq \Sigma_i^*$. We also have that $K = \cap_{i\leq n} P_i^{-1}(\mathcal{E}_i)$.

Let $s \in K_i$. Consider now $s', s'' \in K_i(s, \Sigma_{uc})$ and $\sigma \in \Sigma_{i,uc}$ be such that $P_i(s') = P_i(s'')$, $s'\sigma \in P_i^{-1}(\mathcal{L}(G_i))(s, \Sigma_{uc})$ and $s''\sigma \in K_i(s, \Sigma_{uc})$. We have to show that $s'\sigma \in K_i(s, \Sigma_{uc})$ or equivalently that $ss'\sigma \in K_i$.

As $ss' \in K_i$ and $K_i \subseteq K$, we have $ss' \in K = \cap_{j\leq n} P_j^{-1}(\mathcal{E}_j)$, which entails that $\forall j \leq n$, $ss' \in P_j^{-1}(\mathcal{E}_j)$. Thus $P_j(ss') \in \mathcal{E}_j$. Moreover, $\sigma \in \Sigma_i \setminus \Sigma_s$, from which we can deduce that $\forall j \neq i,\ P_j(ss'\sigma) = P_j(ss')$ and that $P_j(ss'\sigma) \in \mathcal{E}_j$. Thus, $\forall j \neq i$, $ss'\sigma \in P_j^{-1}(\mathcal{E}_j)$.

Let us now show that $ss'\sigma \in P_i^{-1}(\mathcal{E}_i)$. By hypothesis, we have $ss''\sigma \in K_i$. Hence $ss''\sigma \in K$. This entails that $ss''\sigma \in P_i^{-1}(\mathcal{E}_i)$. By applying Lemma 1, we obtain that $sP_i(s''\sigma) \in P_i^{-1}(\mathcal{E}_i)$. However, $\sigma \in \Sigma_i$, hence $sP_i(s'')\sigma \in P_i^{-1}(\mathcal{E}_i)$. Moreover, $P_i(s'') = P_i(s')$; thus $sP_i(s')\sigma \in P_i^{-1}(\mathcal{E}_i)$. Hence $P_i(sP_i(s')\sigma) \in \mathcal{E}_i$ and as $P_i(sP_i(s'\sigma)) = P_i(ss'\sigma)$, we obtain that $P_i(ss'\sigma) \in \mathcal{E}_i$, which entails that $ss'\sigma \in P_i^{-1}(\mathcal{E}_i)$.

Finally $\forall 1 \leq j \leq n,\ ss'\sigma \in P_j^{-1}(\mathcal{E}_j)$. Hence $ss'\sigma \in K$. However, $ss'\sigma \in P_i^{-1}(\mathcal{L}(G_i))$ by hypothesis, thus $ss'\sigma \in K_i$. And finally $s'\sigma \in K_i(s, \Sigma_{uc})$. Hence the result. $\diamond$

The previous proposition states that whenever the specification is separable then our methodology offers an alternative way to the one of [11] to compute $(K \cap \mathcal{L}(G))^{\uparrow c}$. Indeed, the solution of [11] that gives access to a set of decentralized supervisors acting upon each local component of the plant whereas, in our case, the result is a centralized supervisor (See Section 4.3 above).

A contrario, the next example shows that a language that respects the G-Observability, may be not separable.

Example 3 *Let $G = G_1 \parallel G_2$ with $\mathcal{L}(G_1) = \overline{\{a_1.uc_1\}}$ and $\mathcal{L}(G_2) = \overline{\{a_2.uc_2\}}$. We have $\Sigma_{1,uc} = \{uc_1\}$ and $\Sigma_{2,uc} = \{uc_2\}$. Let $K = \overline{\{a_1.uc_1, a_2.uc_2\}}$ be a prefix-closed language over $\Sigma = \Sigma_1 \cup \Sigma_2$. One can check that K verifies the G-observability condition of Theorem 4. However, K is not separable w.r.t. $\{\Sigma_1, \Sigma_2\}$, since $a_1 \in P_1(K)$ and $a_2 \in P_2(K)$, thus $a_1a_2 \in P_1(K) \parallel P_2(K)$ but $a_1a_2 \notin K$.* $\diamond$

4.3 The Supervisor acting upon G.

Let us now describe the way a supervisor can be extracted from the previously computed languages and how it can act upon G in order to achieve the control objective K. With the notations of Theorem 2, $\bigcap_{i\leq n} K_i^{\uparrow pc}$ is controllable with respect to Σ_{uc} and $\mathcal{L}(G)$. However, it is not of interest to perform the intersection between these languages and to derive a supervisor from the result (all the computational advantages of our method would be lost). Following the concept of modularity described in [12], the supervisor $\mathcal{S}$ will be seen as an oracle taking its decision according to the history of the system and the so-called pc-supervisors $\mathcal{S}_i^{pc}$ derived from $K_i^{\uparrow pc}$. The supervisor architecture is summarized in Figure 5.

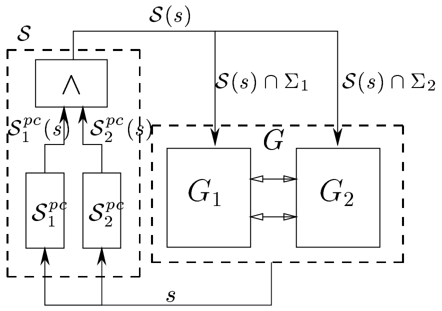


Figure 5: Supervision Scheme

From each $K_i^{\uparrow pc}$, we derive a "supervisor" $\mathcal{S}_i^{pc}$, which after a trace of G (which is also a trace of $P_i^{-1}(\mathcal{L}(G_i))$), delivers the set of events that extend s in a trace of $K_i^{\uparrow pc}$ (each of these pc-supervisor ensures on $P_i^{-1}(\mathcal{L}(G_i))$ the partial controllability property w.r.t. K and $P_i^{-1}(\mathcal{L}(G_i))$). Further following the modularity concept, the set of allowed events is given by $\mathcal{S}(s) = \mathcal{S}_1^{pc}(s) \cap \cdots \cap \mathcal{S}_n^{pc}(s)$. Finally, the sub-set of events that is allowed in G_i is given by $\mathcal{S}(s) \cap \Sigma_i$.

5 Conclusion

In this paper, we have investigated the Supervisory Control of Concurrent Discrete Event Systems. In particular, we proposed a modular method allowing to compute the supremal language of a specification K controllable w.r.t. to the plant G. From the plant G and each of its components G_i, we derive a set of approximations $\mathcal{L}(G_i^{-1})$ and we ensure by control that each of these approximations respects a new language property, called partial controllability condition that depends on K. It is then shown that whenever the original specification respects some conditions (either $K \subseteq \mathcal{L}(G)$, or K is G-observable) then a centralized supervisor can be extracted from the controlled approximations in such a way that the behavior of the controlled plant corresponds to the supremal controllable language contained in the one of the specification. Let us now emphasize some points that we did not mention so far but that are easy to deduce from the theorem (2,3,4).

- If the specification is given in a modular way e.g. $K \cap K'$, then in the case of prefix-closed languages, the modularity results in [12] together with the results of Theorem 3, ensures that
$$\bigcap_{i \le n} K_i^{\uparrow pc} \cap \bigcap_{i \le n} K_i'^{\uparrow pc} = ((K \cap K')) \cap \mathcal{L}(G)^{\uparrow c}$$
as far as the conditions of Theorems 3 or 4 are satisfied by the two specifications.

- If one want to change a component of G, e.g. replacing G_i by G_i', then as far as G_i' is expressed using the same alphabet as the one of G_i with the same partitioning between the controllable/uncontrollable event, then it is sufficient to recompute $K_i'^{\uparrow pc} = (K \cap P_i{}^{-1}(\mathcal{L}(G_i'))^{\uparrow pc}$ in order to obtain the new supervisor (note that only the conditions referring to G_i' has to be (re)-checked).

  Hence this methodology is suitable for reconfigurable plants.

So far we have been interested in the control of plant for prefix-closed specification. We are currently looking for results ensuring that the controlled plant is non-blocking while still avoiding the computation of the whole state space. Another point of interest would be to extend theses techniques to the hierarchical model described in [4].

References

[1] S. Abdelwahed and W. Wonham. Supervisory control of interacting discrete event systems. In *41th IEEE Conference on Decision and Control*, pages 1175–1180, Las Vegas, USA, December 2002.

[2] C. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems*. Kluwer Academic Publishers, 1999.

[3] M.H. deQueiroz and J.E.R. Cury. Modular supervisory control of large scale discrete-event systems. In *Discrete Event Systems: Analysis and Control. Proc. WODES'00*, pages 103–110. Kluwer Academic, 2000.

[4] B. Gaudin and H. Marchand. Modular supervisory control of asynchronous and hierarchical finite state machines. In *European Control Conference, ECC 2003*, Cambridge, UK, September 2003.

[5] S. Jiang and R. Kumar. Decentralized control of discrete event systems with specializations to local control and concurrent systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 30(5):653–660, October 2000.

[6] K. Åkesson, Flordal, and M. Fabian. Exploiting modularity for synthesis and verification of supervisors. In *Proc. of the IFAC*, barcelona, Spain, July 2002.

[7] R.J. Leduc, B.A. Brandin, W.M. Wonham, and M. Lawford. Hierarchical interface-based supervisory control: Serial case. In *Proc. of 40th Conf. Decision Contr.*, pages 4116–4121, December 2001.

[8] P. J. Ramadge and W. M. Wonham. The control of discrete event systems. *Proceedings of the IEEE; Special issue on Dynamics of Discrete Event Systems*, 77(1):81–98, 1989.

[9] K. Rohloff and S. Lafortune. The control and verification of similar agents operating in a broadcast network environment. In *42nd IEEE Conference on Decision and Control*, Hawaii, USA, December 2003.

[10] J. N. Tsitsiklis. On the control of discrete event dynamical systems. *Mathematics of Control Signals and Systems*, 2(2):95–107, 1989.

[11] Y. Willner and M. Heymann. Supervisory control of concurrent discrete-event systems. *International Journal of Control*, 54(5):1143–1169, 1991.

[12] W. M. Wonham and P. J. Ramadge. Modular supervisory control of discrete event systems. *Mathematics of Control Signals and Systems*, 1:13–30, 1988.