CDC02-REG0793
Supervisory Control Problems of Hierarchical Finite State Machines

H. Marchand, B. Gaudin
VerTeCs Team, Irisa, Campus Universitaire de Beaulieu, 35042 Rennes, France,
E-mail: {hmarchan,bgaudin}@irisa.fr

Abstract
The situation under consideration is that of a given Discrete Event System (DES), whose behavior has to be modified by means of a feedback control (named Supervisor) in order to achieve a given set of requirements that the initial DES did not satisfy. To do so, the DES is modeled as a Hierarchical Finite State Machine (HFSM). Further, instead of translating the HFSM to ordinary state machines and using classical synthesis tools on the resulting FSM, we here present algorithms that solve the Non-Blocking Supervisory Control Problem (for a particular case of forbidden state avoidance problem) as well as the Optimal Control Problem without expanding the HFSM.

Key words : Discrete Event Systems, Supervisory Control Problem, Hierarchical Structures, Optimal Control

1 Introduction

The theory of automatic supervisor synthesis of DES has been the subject of numerous studies since the beginning of the 80's (see e.g. [5]). Given a plant (P) and a specification of the expected behavior (S), the control of the plant is performed by inhibiting some events belonging to a set of controlled events while the other events cannot be prevented from occurring (they are said to be uncontrollable) in such a way that the behavior of the controlled plant is included (in the sense of language) in (S). In many applications and control problems, Finite State Machine (FSMs) are the starting point to model fragments of a system, which usually consists of the composition of many different sub-systems. Each of these subsystems consists of different sub-systems. Further, the standard way of applying the supervisor synthesis methodology to hierarchical state machines is by translating them to ordinary state machines and by using classical synthesis tools on the resulting FSM. However, knowing that the synthesis algorithms are polynomial in the number of states of the systems and that the number of states of the global system grows exponentially with the number of parallel and nested sub-systems, it seems important to design algorithms that perform controller synthesis without expanding the system by taking advantage of the hierarchy of the system.

Some techniques based on model aggregation methods [8] have been proposed to deal with hierarchical control problems. However, in this paper, we are more interested in applying existing techniques to a multi-level hierarchy model. The notion of Hierarchical state machines was first popularized by Harel in [4], who introduced the STATECHARTS model. STATECHARTS are a very rich graphical specification formalism allowing (among others) the nesting of state machines (inducing an implicit hierarchy), the orthogonality of state machine (inducing parallelism between state machines) and re-usability of components in different contexts. For supervisory controller and verification purposes, Brave and Heimann in [2] introduced Hierarchical State Machines which constitute a simplified version of the STATECHARTS. Compared to the classical state machines, they add orthogonality and hierarchy features. Further, Alur and al. introduced the hierarchical Kripke Structure in [1] allowing re-usability of components (called structures) in different contexts. The Hierarchical Finite State Machines (HFSM) we are considering can be characterized by a collection of nested structures $\langle K_1, \dots, K_n \rangle$, where K_1 represents the top level of the HFSM. At an intermediate level, the structure K_i can be seen as an FSM, in which states can be either ordinary states or super-states b which are constituted by a set of structures running in parallel. Based on this model, we define an algorithm that solves the Non-Blocking Supervisory Control Problem for the case of the invariance as well as the Optimal Control problem without expanding the HFSM. The result is an HFSM that has the same behavior as the one that would have been obtained by expanding the HFSM and by applying the classical algorithm on the resulting FSM.

2 Preliminaries

The basic structures considered are Finite State Machines (FSMs) defined by a 5-tuple $G = \langle \Sigma, \mathcal{X}, \mathcal{X}_o, x_f, \delta \rangle$, where Σ is a finite alphabet. $\mathcal{X}$ is a finite set of states, $\mathcal{X}_o \subseteq \mathcal{X}$ is the set of initial states, whereas x_f is the final (marked) state, $\delta : \Sigma \times \mathcal{X} \longrightarrow \mathcal{X}$ is a partial transition function. 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, $\delta(s, x)$ denotes the state reached by taking the sequence of events defined by trace s from state x in machine G. $\delta(x)$ denotes the active event set of x. Similarly, $\delta^{-1}(x)$ denotes the set of events that lead to x. The behavior of the system is described by a pair of languages $\mathcal{L}(G) \subseteq \Sigma^*$ and $\mathcal{L}_m(G)$. The prefix-closed language $\mathcal{L}(G)$ is the language generated by G. Similarly, the language $\mathcal{L}_m(G)$ corresponds to the marked behavior of the FSM G, i.e., the set of trajectories of the system ending in x_f, which can be thought of as the state reached when a given task is completed. An FSM G is said to be *blocking* if $\mathcal{L}(G) \neq \overline{\mathcal{L}_m(G)}$ and non-blocking if $\mathcal{L}(G) = \overline{\mathcal{L}_m(G)}$, where $\overline{K}$ denotes the prefix closure of the language K. It can be shown [3] that G is non-blocking whenever it is *trim* with respect to $\mathcal{X}_o$ and x_f (i.e., all the states of G are reachable from on $\mathcal{X}_o$ and co-reachable to the final state x_f).

We now introduce the notion of submachines of an FSM [3].

Definition 1 *A FSM $H = \langle \Sigma_H, \mathcal{X}_H, \mathcal{X}_{H_o}, x_f, \delta_H \rangle$ is a submachine of G, denoted $H \subseteq G$, if $\Sigma_H \subseteq \Sigma$, $\mathcal{X}_H \subseteq \mathcal{X}$, $\forall \sigma \in \Sigma_H, x \in \mathcal{X}_H$ $\delta_H(\sigma, x)! \Rightarrow (\delta_H(\sigma, x) = \delta(\sigma, x))$.*

In the sequel, we will denote by $G(x, x')$ the trim submachine of G initialized in state $x \in \mathcal{X}$ and ending in x' (or simply $G(x)$ when the final state is x_f). We now define the asynchronous product of two FSMs. This operation will be intensively used in the sequel to combine the different FSMs involved in the specification of the plant we want to control.

Definition 2 *Consider two FSMs,* $G_1 = \langle \Sigma_1, \mathcal{X}_1, \mathcal{X}_{o_1}, x_{f_1}, \delta_1 \rangle$ *and* $G_2 = \langle \Sigma_2, \mathcal{X}_2, \mathcal{X}_{o_2}, x_{f_2}, \delta_2 \rangle$*, with* $\Sigma_1 \cap \Sigma_2 = \emptyset$*. The asynchronous product of* G_1 *and* G_2*, noted* $G_1 \| G_2$*, is another FSM* $\langle \Sigma_{12}, \mathcal{X}_{12}, \mathcal{X}_{o_{12}}, x_{f_{12}}, \delta_{12} \rangle$*, such that* $\Sigma_{12} = \Sigma_1 \cup \Sigma_2$*,* $\mathcal{X}_{12} = \mathcal{X}_1 \times \mathcal{X}_2$*, the new set of initial states is given by* $\mathcal{X}_{o_{12}} = \mathcal{X}_{o_1} \times \mathcal{X}_{o_2}$ *and the final state by* $x_{f_{12}} = \langle x_{f_1}, x_{f_2} \rangle$*. The partial transition function* δ_{12} *is defined by:*

$$\delta_{12}(\sigma, \langle x_1, x_2 \rangle) = \begin{cases} \langle \delta_1(\sigma, x_1), x_2 \rangle \text{ if } \sigma \in \Sigma_1 \text{ and } \delta_1(\sigma, x_1)! \\ \langle x_1, \delta_2(\sigma, x_2) \rangle \text{ if } \sigma \in \Sigma_2 \text{ and } \delta_2(\sigma, x_2)! \\ \text{Undefined otherwise} \end{cases} \quad (1)$$

3 The Hierarchical Finite State Machine

Now, we add new features to the definition of an FSM, namely the hierarchy and the re-usability of components. From now on, states (called super-states) of an FSM can be other FSMs. Informally, the meaning of such a hierarchical definition is obtained by recursively substituting each super-state by a set of FSMs running in parallel. Some of these FSMs, called *structures*, may appear repeatedly in different contexts; however, it is assumed that no interaction exists between parallel structures. Such a model is called Hierarchical Finite State Machine (HFSM). The following example illustrates this informal definition.

Example 1 *The HFSM is given by 4 structures* $\langle K_1, K_2, K_3, K_4 \rangle$*. It is represented as in Figure 1.* K_1 *is the upper structure. It comprises 2 super-states* b_1 *and* b_2*.*

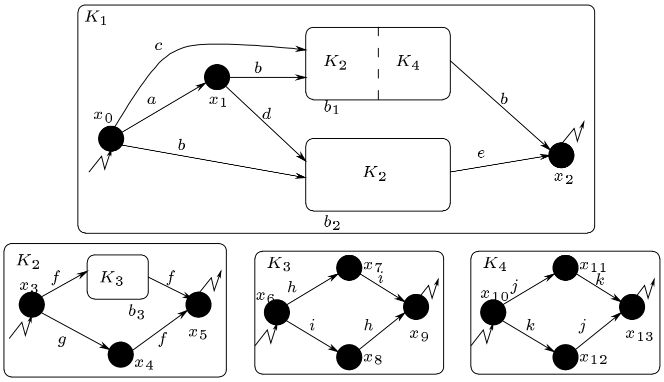

Figure 1: An Example

The super-states b_1 *corresponds to a particular sub-behavior modeled by the couple* K_2 *and* K_4 *running in parallel, whereas* b_2 *simply corresponds to the sub-behavior of* K_2*. At a lower level,* K_3 *is a flat FSM contained in the structure* K_2*.*

These remarks lead us to the following definition of a Hierarchical Finite State Machine. First we define the notion of structures.

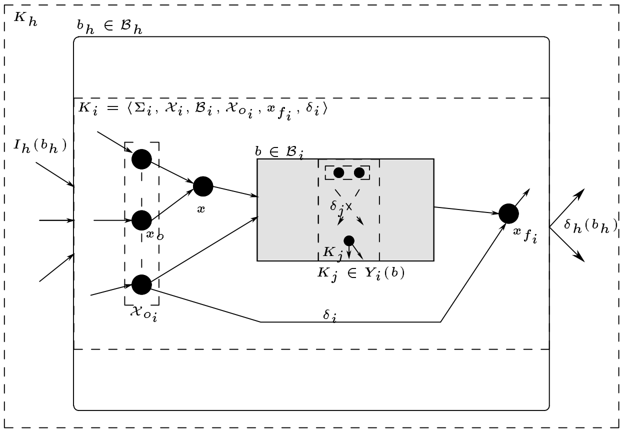

Figure 2: An intuitive view of an HFSM

Definition 3 *A structure* K *is a tuple* $\langle \Sigma, \mathcal{X}, \mathcal{B}, \mathcal{X}_o, x_f, \delta \rangle$*, where* $\mathcal{X}$ *is a set of atomic states,* $\mathcal{X}_o \subseteq \mathcal{X}$ *is the set of initial states and* $x_f \in \mathcal{X}$ *is the final state of* K*.* $\mathcal{B}$ *is the set of super-states of* K*.* δ *is the partial transition function of* K *defined over* $\Sigma \times \{\mathcal{X} \cup \mathcal{B}\} \to \{\mathcal{X} \cup \mathcal{B}\}$*.*

In the following we will denote by K^A the structure K seen as an FSM (i.e. when the super-states are considered as atomic states). The notion of structure allows us to define the notion of Hierarchical Finite State Machines.

Definition 4 *A Hierarchical Finite State Machine* $\mathcal{K}$ *is given by a tuple* $(\langle K_1, \ldots, K_n \rangle, R)$*, where* $\forall 1 \leq i \leq n$*,* $K_i = \langle \Sigma_i, \mathcal{X}_i, \mathcal{B}_i, \mathcal{X}_{o_i}, x_{f_i}, \delta_i \rangle$ *is a structure (with* $\mathcal{B}_n = \emptyset$ *and* $\Sigma_i \cap \Sigma_j = \emptyset$ *for all* $i \neq j$*), and* R *is a set* $(Y_i, I_i)_{i \in J}$*, with* $J = \{1 \leq i \leq n \mid \mathcal{B}_i \neq \emptyset\}$*.* R *characterizes the hierarchy between the structures. For all* $i \in J$*,* (Y_i, I_i) *is defined by :*

- $Y_i : \mathcal{B}_i \longrightarrow 2^{\langle K_{i+1}, \ldots, K_n \rangle}$ *is a function which maps each super-state* $b \in \mathcal{B}_i$ *on a set of structures* K_j*, with* $j > i$*. We use* J_b^i *(or* J_b *for short) as* $\{j \leq n\} \mid K_j \in Y_i(b)\}$*. The structures of* $Y_i(b)$ *behave asynchronously (see Def. 2).*
- I_i *is an Input function that gives access to the set of initial states that are reached when triggering an event that makes the system evolve into a super-state* b*.* $\forall b \in \mathcal{B}_i$*,* $I_i(b)$ *is a function defined over* $\Pi_{j \in J_b} \mathcal{X}_{o_j} \to 2^{\Sigma_i}$*. Given a super-state* $b \in \mathcal{B}_i$*, and* $x_o = < x_{o_1}, \ldots, x_{o_{\|J_b\|}} >$ *a tuple of initial states,* $I_i(b)(x_o)$ *corresponds to the events that make the system go from its current state into* x_o*.*[1]

K_1 will be called the *root* of $\mathcal{K}$, whereas the structures K_i such that $\mathcal{B}_i = \emptyset$ will be called the *leaves* of $\mathcal{K}$ and the structures (recursively) involved in the definition of another structure K_i will be called the sons of K_i.

Given an HFSM $\mathcal{K} = (\langle K_1, \ldots, K_n \rangle, R)$, and a particular structure, say K_i, when an event $\sigma \in \delta_i^{-1}(b)$ is triggered, all the structures of $Y_i(b)$ are simultaneously activated and entered in one of their initial states according to $I_i(b)$. After entering a super-state, the different structures evolve

[1] For example, $I_1(b_1) = (\langle x_3, x_{10} \rangle \mapsto \{b, c\})$, whereas $I_1(b_2) = (\langle x_3 \rangle \mapsto \{b, d\})$.

asynchronously following Definition 2. However, in order to evolve out of a super-state b, there is synchronization between the different structures of $Y_i(b) = (K_i)_{i\in J_b}$ on their final state. Hence, an event $\sigma \in \delta_i(b)$ can be triggered in a super-state b whenever each substructure of $Y_i(b)$ is in its corresponding final state (*i.e.*, there is no preemption).

In order to be able to perform control on an HFSM, we need to make some assumptions on it.

Assumption 1 *Let $b \in \mathcal{B}_i$ be a super-state of K_i, and $(K_j)_{j\in J_b} = Y_i(b)$ be the corresponding structures attached to b. Then, $\{I_i(b)(X_o)\}_{X_o \in \Pi_{j\in J_b}\mathcal{X}_{o_j}}$ is a partition of $\delta_i^{-1}(b)$.*

This assumption ensures that each event leading to b is taken into account, and that there is a unique tuple of initial states that has σ as input event (i.e. entering the super-state is deterministic).

Assumption 2 *Two structures of a super-state cannot have common sons.*

This assumption ensures that a structure can not be in parallel with itself. From a control point of view, it means that it will be possible to use exactly one supervisor in order to control a structure.

Expanded HFSM. Given an HFSM $\mathcal{K} = (\langle K_1, \dots, K_n\rangle, R)$, we can make correspond an FSM. It is recursively obtained as follows. First, the leaves of $\mathcal{K}$ are already expanded. Now assume the structures $K_{i+1}, \dots, K_n$ have already been expanded to FSMs and denoted by $\mathcal{K}_j^F = \langle \Sigma_j^F, \mathcal{X}_j^F, \mathcal{X}_{o_j}^F, x_{f_j}^F, \delta_j^F\rangle$, $i+1 \le j \le n$. We now expand the structure K_i itself. To each super-state b of K_i, we associate its corresponding FSM K_b^F obtained by performing the asynchronous product between each expanded structures of $Y_i(b)$, i.e. $K_b^F = \langle \Sigma_b^F, \mathcal{X}_{o_b}^F, x_b^F, x_{f_b}^F, \delta_b^F\rangle = \|_{j\in J_b}\mathcal{K}_j^F$. To expand the structure K_i, we replace each super-state b of $\mathcal{B}_i$ by its corresponding FSM K_b^F (the states will be denoted $[b, \langle x_1^F, \dots, x_{\|J_b\|}^F\rangle]$) and we connect the initial states of K_b^F to the states of K_i according to I_i (resp. for the final state). The result is an FSM, denoted by $\mathcal{K}_i^F$. Such an FSM is called the *expanded structure* of K_i. We denote by $\mathcal{K}^F$ the expanded structure of $\mathcal{K}$ (it is equal to $\mathcal{K}_1^F$). One can remark that the behavior of $\mathcal{K}$ is equal to $\mathcal{L}(K^F)$.

Sub-behavior of an HFSM. In the sequel, we wish to apply some control to the original system $\mathcal{K}$ in order to ensure a given control objective. In other words, we wish to reduce the system $\mathcal{K}$ to a particular behavior. This leads us to define the two following notions of a submachine of an HFSM.

Definition 5 *$\mathcal{H} = (\langle H_1, \dots, H_n\rangle, R')$ with $R' = (Y'_{H_i}, I'_{H_i})_{i\in\{i|\mathcal{B}_i\neq\emptyset\}}$ is a sub-HFSM of $\mathcal{K} = (\langle K_1, \dots, K_n\rangle, R)$ (noted $\mathcal{H} \subseteq \mathcal{K}$), whenever*

- *$\mathcal{H}$ is an HFSM and $\forall i,\ H_i \subseteq K_i$ (i.e. $H_i^A \subseteq K_i^A$ and $\mathcal{B}_{H_i} \subseteq \mathcal{B}_{K_i}$). $\forall i, H_i$ is supposed to be trim.*
- *R' respects the following conditions. $\forall 1 \le i \le n$, such that $\mathcal{B}_{H_i} \neq \emptyset$, $\forall b \in \mathcal{B}_{H_i}$,*
  1. *$Y_{H_i}(b) = (H_j)_{j\in J_b}$, whenever $Y_i(b) = (K_j)_{j\in J_b}$*
  2. *$I_{H_i}(b)$ is the sub-function of $I_i(b)$ restricted according to δ_{H_i}. In other words, $\forall b \in \mathcal{B}_{H_i}$,*
  $$\forall x_o \in \prod_{j\in J_b} \mathcal{X}_{H_{o_j}},\ I_{H_i}(b)(x_o) = I_i(b)(x_o) \cap \delta_{H_i}^{-1}(b).$$

Note that, in general, some structures of $\mathcal{H}$ may be useless (i.e. they are no more associated to a super-state).

In some situation, it will be useful to reduce an HFSM to a particular level (i.e. considering a structure K_i of $\mathcal{K}$ as the new root of the HFSM).

Definition 6 *Let $\mathcal{K} = (\langle K_1, \dots, K_n\rangle, R)$ be an HFSM and K_i a structure. Then $\mathcal{K}_i$ corresponds to the HFSM $\mathcal{K}_i = (\langle K_i, K_{j_1}, \dots, K_{j_l}\rangle, R_{|\{i,j_1,\dots,j_l\}})$, where K_i is seen as the root of the HFSM and the K_{j_k} are the sons of K_i.*

4 Supervisory Control Problem of HFSM

The idea of the Supervisory control theory is that the plant models the uncontrolled behavior, which is not fully satisfactory and must be "restricted" by means of a controller called a *supervisor* [5]. However, not all but only a part of the events can be disabled by the supervisor. Therefore, some of the events are said to be uncontrollable, i.e., their occurrence cannot be prevented by a controller, while the others are controllable (see [3] for more details). In this regard, for an HFSM $\mathcal{K} = (\langle K_1, \dots, K_n\rangle, R)$, the alphabets Σ_i can be partitioned as $\Sigma_i = \Sigma_{i_c} \cup \Sigma_{i_{uc}}$, where Σ_{i_c} and $\Sigma_{i_{uc}}$ represent the set of controllable events and the set of uncontrollable events.

4.1 Review of the Supervisory Control Problem for FSM

Assume a plant G is given and modeled as an FSM and a set of states E, we recall how to synthesize a supervisor that will ensure the reachability of the final state while remaining in E. Knowing that some events are uncontrollable, we first recall the definition of *controllable submachine* [5].

Definition 7 *Let G be a FSM and H be a submachine of G, then H is controllable w.r.t. G and Σ_{uc}, whenever $\forall x \in \mathcal{X}_H \subseteq \mathcal{X}, \forall\sigma \in \Sigma_{uc}, \delta(\sigma, x)! \Rightarrow \delta_H(\sigma, x)!$.*

This condition imposes that any transition that needs to be disabled in G to generate H needs to be controllable. Based on Definition 7, we can state the following proposition (its proof is straightforward).

Proposition 1 *Let G_1, G_2 be two FSMs, s.t. $G_i = \langle\Sigma_i, \mathcal{X}_i, \mathcal{X}_{o_i}, x_{f_i}, \delta_i\rangle$ with $\Sigma_1 \cap \Sigma_2 = \emptyset$, and $H_1 \subseteq G_1$, $H_2 \subseteq G_2$. If H_1 (resp. H_2) is controllable with respect to G_1 and $\Sigma_{1_{uc}}$ (resp. G_2 and $\Sigma_{2_{uc}}$), then $H_1\|H_2$ is controllable w.r.t. $G_1\|G_2$ and $\Sigma_{1_{uc}} \cup \Sigma_{2_{uc}}$.* $\diamond$

A supervisor $\mathcal{S}$ is given by a function $S : \mathcal{X} \to 2^{\Sigma_c}$, delivering the set of actions that are disabled in state x of G by the control[2], and the new set of valid initial states $\mathcal{X}'_o \subseteq \mathcal{X}_o$ (it could be the case, that in order to ensure an objective, we need to reduce $\mathcal{X}_o$). Write $\mathcal{S}/G$ for the system, consisting of

[2] In general, S is a function from $\mathcal{L}(G)$ into 2^{Σ}.

the initial plant G controlled by the supervisor $\mathcal{S}$. The Supervisory Control Problem (SCP) is then the following: *given G and E a set of states, the problem is to build a supervisor $\mathcal{S}$ such that (1) $\mathcal{S}/G$ is controllable (2) all the trajectories of $\mathcal{S}/G$ eventually lead to the final state, (3) the traversed states belong to E and (4) $\mathcal{S}/G$ is the most permissive solution.* In the sequel, we will be more interested in the computation of $\mathcal{S}/G$ rather than in the computation of the supervisor S itself, since one can easily extract $\mathcal{S}$ from $\mathcal{S}/G$. The standard algorithm (called RSCP) to compute $\mathcal{S}/G$ is an iterative algorithm starting with $H \subseteq G$ the FSM obtained by removing from G all the states that do not belong to E. The iterative procedure consists of (i) removing states that violate the controllability condition, and (ii) removing states that are not reachable. Call $H^{\uparrow} \subseteq H$ the result. If $H^{\uparrow}$ is not reduced to the empty FSM, then $H^{\uparrow}$ is the greatest controllable submachine of G that both ensures the invariance of E and the reachability of x_f.

Let us now present a result that will be useful in the HFSM framework. First we need the following lemma.

Lemma 1 *Let G be an FSM and $H \subseteq G$, then the greatest controllable submachine of H w.r.t. G and Σ_{uc} contains all the other controllable submachines of H w.r.t. G and Σ_{uc}.* ⋄

Based on Lemma 1, we can prove the following property:

Proposition 2 *Let G_1 and G_2 be two FSMs, and H_1, H_2 be two submachines of G_1 and G_2. Let $H_1^{\uparrow}$ (resp. $H_2^{\uparrow}$) be the greatest controllable submachine of H_1 (resp. H_2) w.r.t. G_1 and $\Sigma_{1_{uc}}$ (resp. G_2 and $\Sigma_{2_{uc}}$), then $H_1^{\uparrow} \| H_2^{\uparrow} = (H_1 \| H_2)^{\uparrow}$, where $(H_1 \| H_2)^{\uparrow}$ corresponds to the greatest controllable submachine of $H_1 \| H_2$ w.r.t $G_1 \| G_2$ and $\Sigma_{1_{uc}} \cup \Sigma_{1_{uc}} = \Sigma_{uc}$.*

Proof : Let us denote by $H^{\uparrow}$ (resp $H_{\uparrow}$) the FSM $H_1^{\uparrow} \| H_2^{\uparrow} = \langle \Sigma_{12}, ..., \delta_{12}^{\uparrow} \rangle$ (resp. $(H_1 \| H_2)^{\uparrow} = \langle \Sigma_{12}, ..., \delta_{\uparrow_{12}} \rangle$). From Proposition 1, $H^{\uparrow}$ is a controllable submachine of $G_1 \| G_2$. Therefore, according to Lemma 1, $H^{\uparrow} \subseteq H_{\uparrow}$ as this FSM is the greatest controllable submachine of $H_1 \| H_2$. Now, assume that $H_{\uparrow} \not\subseteq H^{\uparrow}$. It means that there exists a state $\langle x_1, x_2 \rangle$ and a controllable event σ such that $\delta_{\uparrow_{12}}(\sigma, \langle x_1, x_2 \rangle)!$ and $\delta_{12}^{\uparrow}(\sigma, \langle x_1, x_2 \rangle)$ is not defined. Assume $\sigma \in \Sigma_1$, then according to the algorithm RSCP there exists a path $s \in \Sigma_{1_{uc}}^*$ such that $\delta_1(\sigma s, x_1)$ is a forbidden state[3] (because H_1 is controllable w.r.t. G_1 and $\Sigma_{1_{uc}}$). However, according to Definition 2, σs is also an admissible path which makes $H_1 \| H_2$ evolves into a forbidden state. Therefore this event should have been also removed in $\delta_{\uparrow_{12}}$. Hence the result. ⋄

4.2 The controller synthesis methodology for HFSM

We now extend these algorithms for the case of HFSMs. But first, we need to adapt the controllability definition:

Definition 8 *Let $\mathcal{H} = (\langle H_1, \dots, H_n \rangle, R')$, and $\mathcal{K} = (\langle K_1, \dots, K_n \rangle, R)$ be two HFSMs, then $\mathcal{H}$ is controllable w.r.t. $\mathcal{K}$, whenever $\mathcal{H}$ is a sub-HFSM and $\forall 1 \leq i \leq n$, H_i^A is controllable w.r.t. to K_i^A and $\Sigma_{i_{uc}}$.*

With this definition, we can state the following proposition.

Proposition 3 *If $\mathcal{H}$ is a controllable sub-HFSM of $\mathcal{K}$, then $\mathcal{H}^F$[4] is controllable w.r.t. $\mathcal{K}^F$ and $\Sigma_{uc} = \cup_i \Sigma_{i_{uc}}$.*

[3] By forbidden states, we mean the states of G_i that does not belong to H_i.
[4] $\mathcal{H}^F$ corresponds to the expanded FSM of $\mathcal{H}$

Proof : The proof proceeds by induction. The base case is obvious since the leaves of the HFSM are also FSMs (i.e. $H_j = H_j^F$). Now, assume that $\mathcal{H}_{i+1}, \dots \mathcal{H}_n$ are controllable; by induction hypothesis, their corresponding FSMs $\mathcal{H}_{i+1}^F, \dots, \mathcal{H}_n^F$ are controllable. We will show that if $\mathcal{H}_i$ is controllable then $\mathcal{H}_i^F$ also is. Let x be a reachable state in both $\mathcal{H}_i^F$ and $\mathcal{K}_i^F$ and $\sigma_{uc} \in \Sigma_{i_{uc}}$ be an admissible uncontrollable event in this state for the FSM $\mathcal{K}_i^F$. Then depending on x and the reached state, different cases can occur.

- if x is an atomic state of the structure K_i of $\mathcal{K}_i$, then
  - if $\delta_i^F(\sigma_{uc}, x) \in \mathcal{X}_i$, then $\delta_{H_i}^F(\sigma_{uc}, x)$ certainly exists since H_i^A is controllable w.r.t. K_i^A.
  - If $\delta_i^F(\sigma_{uc}, x)$ is of the form $[b, \langle x_{o_1}, \dots, x_{o_{j_b}} \rangle]$, it means that $\mathcal{K}_i$ evolves into a super-state $b \in \mathcal{B}_i$ via the event σ_{uc}. As H_i^A is controllable w.r.t. K_i^A, $\delta_{H_i}(\sigma_{uc}, x)!$ and is equal to b. Hence, as $\mathcal{H}_i \subseteq \mathcal{K}_i$ and according to Assumption 2, we enter super-state b through the unique tuple of initial states $\langle x_{o_1}, \dots, x_{o_{j_b}} \rangle$. Hence, $\delta_{H_i}^F(\sigma_{uc}, x)!$ and is equal to $[b, \langle x_{o_1}, \dots, x_{o_{j_b}} \rangle]$.
- if x is of the form $[b, \langle x_1, \dots, x_{j_b} \rangle]$, which means that $\mathcal{K}_i$ currently evolves inside b.
  - If $[b, \langle x_1, \dots, x_{j_b} \rangle] \in \mathcal{X}_b^F \setminus \{x_{f_b}^F\}$, the result is due to the induction hypothesis and Proposition 1.
  - Assume now that $[b, \langle x_1, \dots, x_{j_b} \rangle] = x_{f_b}^F$. As $\delta_i^F(\sigma_{uc}, [b, \langle x_1, \dots, x_{j_b} \rangle])!$, it means that $\sigma_{uc} \in \delta_i(b)$. Now, as H_i^A is a controllable submachine of K_i^A, we can say that $\sigma_{uc} \in \delta_{H_i}(b)$, which entails that $\delta_{H_i}^F([b, \langle x_1, \dots, x_{j_b} \rangle])!$.

We shown that for all reachable states x of $\mathcal{H}_i^F$, if $\delta_i^F(\sigma_{uc}, x)!$, $\delta_{H_i}^F(\sigma_{uc}, x)$ also exists. Then $\mathcal{H}_i^F$ is controllable w.r.t. $\mathcal{K}_i^F$ and $\Sigma_{i_{uc}}$, and finally $\mathcal{H}^F$ is controllable w.r.t. $\mathcal{K}^F$ and Σ_{uc}. ⋄

The supervisor Now that we have the definition of a controllable sub-HFSM, it is interesting to determine how such a sub-HFSM can be obtained via a supervisor acting upon $\mathcal{K}$.

Definition 9 *Let $\mathcal{K} = (\langle K_1, \dots, K_n \rangle, R)$ be an HFSM then a supervisor $\mathcal{S}$ is given by a set $(S_i, \mathcal{X}'_{o_i})_{i \in [1..n]}$, where S_i is a function $\mathcal{X}_i \cup \mathcal{B}_i \to 2^{\Sigma_{i_c}}$ and $\mathcal{X}'_{o_i}$ corresponds to the new valid initial states of K_i.*

Conceptually, the supervisor $\mathcal{S}$ controlling the plant (i.e. the HFSM) observes the configuration x in which the plant is. Then according to the structures that are active in x, the local supervisors are activated and works as a classical supervisor (namely, it enables/disables events according to the current local state). To conclude this section, let us remark that this definition is consistent with the definition of controllable submachine of an HFSM. This is summarized by Proposition 4.

Proposition 4 *Let $\mathcal{H}$ be a sub-HFSM of $\mathcal{K}$. Then $\mathcal{H}$ is a controllable sub-HFSM of $\mathcal{K}$ if and only if there exists a supervisor $\mathcal{S} = (S_i, \mathcal{X}'_{i_o})_{i \in [1..n]}$ such that $\forall\ 1 \leq i \leq n$,*

$$\mathcal{X}_{H_{i_o}} = \mathcal{X}'_{i_o} \text{ and } \forall x \in \mathcal{X}_i \cup \mathcal{B}_i, \delta_{H_i}(x) = \delta_i(x) - S_i(x) \quad (2)$$

The Supervisory Control Problem. In the sequel, we wish to apply some control to the original system $\mathcal{K}$ in order to avoid a set of *forbidden states*, while preserving the reachability of the final state x_{f_1} (i.e. the final state of the root K_1 of $\mathcal{K}$). First, we need to define what we mean by set of states. Due to the hierarchical description of the system, the states of the obtained expanded structure $\mathcal{K}^F$ can have different forms. They are either atomic states or of the form $[b, < \bullet, \bullet, \dots, \bullet >]$. Each $\bullet$ can have one of the previous forms, and so on, recursively, until all the $\bullet$ are atomic states. Intuitively, these atomic states correspond to a particular configuration of the HFSM, i.e. the states in which the structures are simultaneously at a given instant.

Definition 10 *Let $\mathcal{K}$ be an HFSM and $E = (E_i)_{i \le n}$, with $E_i \subseteq \mathcal{X}_i,\ 1 \le i \le n$ be a set of states. Then, the set of expanded states (or configurations) E^F of the expanded structure $\mathcal{K}^F$ is equal to the set of states x such that at least one of the atomic components of x belongs to E.*

Based on this set of states definition, we now present the algorithm that given an HFSM $\mathcal{K}$ and a state of E will compute the controlled HFSM $\mathcal{K}^\uparrow$, such that $(\mathcal{K}^\uparrow)^F$ is the greatest non-blocking controllable submachine of $\mathcal{K}^F$ such that the *set of forbidden states* E^F is not reachable.

To clarify the notations, we will focus on a two levels HFSM, namely a structure, with super-states b, in which all the structures are classical FSMs. The generalization to an arbitrary number of levels is immediate. Let $\mathcal{K} = (\langle K_1, \dots K_n\rangle, (Y_1, I_1))$ be an HFSM, and let $E = (E_i)_{i \le n}$, with $E_i \subseteq \mathcal{X}_i$ be a set of atomic states we want to avoid and x_{f_1} the final state we want to reach. The corresponding algorithm RSCP_HFSM is:

step 1: $\forall j > 2$, compute the greatest controllable system $K_j^\uparrow = \langle \Sigma_j, \mathcal{X}_j^\uparrow, \mathcal{X}_{o_j}^\uparrow, x_{f_j}, \delta_j^\uparrow\rangle$ w.r.t. K_j, $\mathcal{X}_j \setminus E_j$ and $\Sigma_{j_{uc}}$ according to Algo. RSCP.

step 2: $K_1^\uparrow$ is computed as follows:

1. Let $H_1 = \langle \Sigma_1, \mathcal{X}_1, \mathcal{B}_1', \mathcal{X}_{o_1}, x_{f_1} \delta_1' \rangle$ be the substructure of K_1, and Y_1', I_1', s.t. $\mathcal{B}_1'$ is the largest subset of $\mathcal{B}_1$ such that $\forall b \in \mathcal{B}_1'$, with $Y_1'(b) = (K_j^\uparrow)_{j \in J_b}$ whenever $Y_1(b) = (K_j)_{j \in J_b}$.
   (a) if $\exists K_j^\uparrow \in Y_1'(b)$, s.t. $K_j^\uparrow = \emptyset$, then $b \notin \mathcal{B}_1'$
   (b) $\forall x \in \mathcal{X}_1' \cup \mathcal{B}_1'$, $\forall \sigma \in \Sigma_1$, s.t. $\delta_1(\sigma, x) = b$
   $$\delta_1'(\sigma, x) = \begin{cases} b \text{ if } \sigma \in \displaystyle\bigcup_{X_o \in \Pi_{j \in J_b} \mathcal{X}_{o_j}^\uparrow} \{I_1(b)(X_o)\} \\ \text{Undefined otherwise} \end{cases}$$
   (c) Restrict I_1 to I_1' according to the new transition function δ_1'. (Cf Point 2 of Definition 6).
2. Compute the greatest controllable FSM H of H_1^A w.r.t. K_1^A and $\Sigma_{1_{uc}}$ and $\mathcal{B}_1' \cup (\mathcal{X}_1 \setminus E_1)$. Then $K_1^\uparrow$ is the substructure of H_1 s.t. $(K_1^\uparrow)^A = H$

step 3: Call $\mathcal{K}^\uparrow = (\langle K_1^\uparrow, \dots, K_n^\uparrow\rangle, (Y_1', I_1'))$ the resulting HFSM.

Let us describe some particular points of this algorithm. Let $\mathcal{K}' = (\langle H_1, K_2^\uparrow, ..., K_n^\uparrow\rangle, (Y_1', I_1'))$ be the HSFM obtained after step 2 (1.c) of the algorithm. Then, based on Proposition 2, we can say that for each remaining super-state $b \in \mathcal{B}_1'$, and for the set of remaining initial states of b (say $\mathcal{X}_{o_b}$), $(K_b^\uparrow)^F = \|_{j \in J_b} K_j^\uparrow$ is the supremal controllable FSM of $\mathcal{K}^F(\mathcal{X}_{o_b}, x_{f_b})$ w.r.t. $(E_j)_{j \ge 2}^F$ (i.e. we removed all the forbidden configurations corresponding to the set $(E_j)_{j \ge 2}$, while keeping the less restrictive behavior). Note that it may be the case that $\mathcal{K}'^F$ is not controllable w.r.t. $\mathcal{K}^F$. However, we can state the following lemma:

Lemma 2 *With the preceding notations, the greatest controllable submachine of $\mathcal{K}'^F$ w.r.t. $\mathcal{K}^F$ is equal to $(\mathcal{K}^F)^\uparrow$, where $(\mathcal{K}^F)^\uparrow$ is the controlled FSM of the expanded HFSM $\mathcal{K}^F$ obtained using Algo.* RSCP *for the set of forbidden state $(E_j)_{j \ge 2}^F$.*

Further in the algorithm (step 2 (2)), we remove from H_1, the forbidden states corresponding to the structure K_1 (i.e. the set of state E_1) and we compute its greatest controllable submachine. This shows the following theorem:

Theorem 1 *Let $\mathcal{K} = (\langle K_1, \dots, K_n\rangle, R)$ be an HFSM and $E = (E_j)_{j \ge 1}$ be a set of atomic forbidden states. Let $\mathcal{K}^F$ be the expanded structure of $\mathcal{K}$ and E^F be the corresponding set of forbidden configurations. Then, $(\mathcal{K}^\uparrow)^F = (\mathcal{K}^F)^\uparrow$.*

One can remark that, in order to compute the supervisors, the previous algorithm never expands the HFSM and computes exactly once each controlled structure K_i, and that even if K_i appears in different contexts (i.e. as structure of different super-states), thus reducing the complexity and avoiding the state space explosion. This constitutes one of the main difference with the work of [2], where there is no notion of reusability and no notion of "local" supervisor as in our framework (however, they can handle more complex objectives).

4.3 The Optimal Control Problem

The aim of the optimal control is to generate a controller which constrains the system to a desired behavior according to quantitative and qualitative aspects. This is performed by the addition of quantitative measures in the form of cost functions to capture the fact that some legal behaviors are better than others. In this section, we present how the work of [6] can be applied to our model. Compared to [7], we then adopt a multi-level hierarchy model, whereas the "hierarchical optimal control" presented in [7] is based on model aggregation methods.

4.3.1 Review of optimal control on FSM: In this section, we recall some results for the optimal control of FSM (with a unique initial state) that can be found in [6]. In order to take into account the numerical aspect of the optimal control problem, two cost values are associated to each event of Σ. To this effect, we introduce an occurrence cost function $c_e : \Sigma \to \boldsymbol{R}^+$ and a control cost function $c_c : \Sigma \to \{0, \infty\}$. The control cost function is used to encode the status of the events. The control cost function is infinity for events in Σ_{uc}. The cost functions are then used to introduce a cost on the trajectories of a submachine H of G (note that in [6], the control cost function is not trivial). We introduce $\Sigma_d^G(H,x)$ as the set of disabled events at state x for the system to remain in submachine H of G as well as $\mathcal{X}(s,x)$ the set of states crossed on the way when traversing s.

Definition 11 *Let H be a submachine of G and $\mathcal{L}_m(H)$ be the marked language generated by H, then*

- *For any state $x \in \mathcal{X}_H$ and string $s = \sigma_1^s \sigma_2^s \ldots \sigma_{\|s\|}^s$, such that $\delta_H(s,x)$ exists, the cost of s is given by:*

$$c^g(x,H,s) = \sum_{j=1}^{\|s\|} c_e(\sigma_j^s) + \sum_{x' \in \mathcal{X}(s,x)} \sum_{\sigma \in \Sigma_d^G(H,x')} c_c(\sigma) \quad (3)$$

- *The objective function denoted by $c(.)$ is given by: $c(H) = \sup_{s \in \mathcal{L}_m(H)} c^g(x_{0A}, H, s)$.*

Basically, the cost of a trajectory is the sum of the occurrence costs of the events composing it. If an uncontrollable event is disabled, the cost of a trajectory becomes infinite because the second term of (3) becomes infinity. Finally, $c(H)$ represents the worst case behavior possible in submachine H. We now define the optimization problem.

Definition 12 *For all $x \in \mathcal{X}, H \subseteq G(x)$ is an optimal submachine if $c(H) = \min_{H' \subseteq G(x)} c(H') < \infty$.*

In particular, an optimal solution $H \subseteq G(x_0)$ is an optimal submachine of the plant G and represents a solution of the optimal control problem, which in general has more than one solution. For such a submachine H, $c(H)$ represents the optimal cost (in fact, the worst inevitable cost) necessary to reach x_f from x_0. It means that a submachine with a lower cost could not ensure the accessibility of x_f from x_0. The next theorem ([6]) gives necessary and sufficient conditions for the existence of optimal submachines.

Theorem 2 *An optimal submachine of G exists if and only if there exists a submachine H of G such that H is trim, controllable with no cycles.* $\diamond$

We now introduce the notion of DP-Optimal submachines.

Definition 13 *A submachine $H \subseteq G(x)$ is DP-Optimal if it is optimal and for all $x' \in \mathcal{X}_H$, $H(x')$ is an optimal submachine of $G(x')$.*

If a particular DP-Optimal FSM includes all other DP-Optimal FSMs as submachines, then we call it the *maximal DP-Optimal submachine.* Its existence is given by the following theorem ([6]).

Theorem 3 *If an optimal submachine of G exists, then the unique maximal DP-Optimal submachine of G w.r.t. the final state x_f also exists.* $\diamond$

In [6], the authors provide an algorithm (DP-OPT), that, given an FSM G with a unique final state, constructs the maximal DP-Optimal submachine.

4.3.2 The optimal control of asynchronous FSMs: Given two FSMs G_1 and G_2, the purpose of this section is to show that whenever there exist two (DP-)optimal submachines H_1 and H_2 of G_1 and G_2, then $H_1 \parallel H_2$ is also (DP-)optimal w.r.t. $G_1 \parallel G_2$. In the first part of this section, we will consider FSMs with only one initial state as carried out in [6]. First, we need to introduce the following technical lemmas:

Lemma 3 *Let G_1' and G_2' be submachines of G_1 and G_2, then $c(G_1' \parallel G_2') = c(G_1') + c(G_2')$.* $\diamond$

Note that this lemma is true only because we consider the special case, where the control cost function is trivial (i.e. either equal to 0 or ∞).

Lemma 4 *Let $A \subseteq G_1 \parallel G_2$. Then, there exist $G_1' \subseteq G_1$ and $G_2' \subseteq G_2$, s.t. $c(G_1' \parallel G_2') = c(A)$.* $\diamond$

Using the previous results, we are now able to prove the following property:

Proposition 5 *If H_1 and H_2 are optimal submachines of G_1 and G_2, then $H_1 \parallel H_2$ is an optimal submachine of $G_1 \parallel G_2$. The result is still valid when dealing with DP-Optimality.*

Proof : Let A be an optimal submachine of $G_1 \parallel G_2$. According to Lemma 4, there exist $G_1' \subseteq G_1$ and $G_2' \subseteq G_2$, s.t. $c(A) = c(G_1' \parallel G_2')$. Next, it is easy to show that the worst case cost of $H_1 \parallel H_2$ is minimal among the submachines of G of the form $G_1' \parallel G_2'$ (otherwise, either G_1' or G_2' would have a worst case cost lower than the one of H_1 or H_2). Hence we have $c(A) = c(G_1' \parallel G_2') \geq c(H_1 \parallel H_2)$, which proves that $H_1 \parallel H_2$ is optimal w.r.t. $G_1 \parallel G_2$. The DP-Optimality can be easily deduced from the previous point. $\diamond$

So far, we were interested in composing FSMs having only one initial state. Knowing that the structures of the HFSMs have several initial states, we need to extend these results to FSMs having this property.

Definition 14 *Let G be a FSM, and H a submachine of G such that $\mathcal{X}_{H_o} \subseteq \mathcal{X}_{G_o}$ and $\mathcal{X}_{H_o} \neq \emptyset$. H is (DP-)optimal if $\forall x_0 \in \mathcal{X}_{H_o}$, $H(x_0)$ is (DP-)optimal w.r.t. $G(x_0)$, and $\forall x_0 \in \mathcal{X}_{G_o} \setminus \mathcal{X}_{H_o}$, $G(x_0)$ has no optimal submachine.*

With the preceding notations, consider the maximal DP-Optimal submachine $H(x_o)$ of $G(x_o)$. We denote by $\mathcal{X}_o'$ the set of initial states for which such a submachine exists. It is then easy to show that $H = \oplus_{x_o \in \mathcal{X}_o'}(H(x_o))$[5] is DP-optimal w.r.t. G. Based on this result, we can state the following proposition:

Proposition 6 *Let H_1 and H_2 be DP-optimal submachines of G_1 and G_2, then $H_1 \parallel H_2$ is a DP-optimal submachine of $G_1 \parallel G_2$.* $\diamond$

Proof : We just give here the sketch of the proof. For all $x_{o_i} \in \mathcal{X}_{o_1}'$ and $x_{o_j} \in \mathcal{X}_{o_2}'$, using Proposition 5, we know that $H_{ij} = H_1(x_{o_i}) \parallel H_2(x_{o_j})$ is DP-optimal w.r.t. $G_1(x_{o_i}) \parallel$

[5] $\oplus$ denotes the merge of two FSMs (see [6]).

$G_2(x_{o_j})$. Now, based on the fact that the merge of two DP-Optimal submachines is still a DP-Optimal submachine, we can deduce that $\oplus_{ij} H_1(x_{o_i}) \parallel H_2(x_{o_j}) = H_1 \parallel H_2$ is a DP-Optimal of $G_1 \parallel G_2$. Consider now a state $x_o = \langle x_{o_1}, x_{o_2}\rangle \notin \mathcal{X}'_{o_1} \times \mathcal{X}'_{o_2}$. Assume $x_{o_1} \notin \mathcal{X}'_{o_1}$. It entails that there exists no DP-Optimal submachine of $G_1(x_{o_1})$, and that $\forall H(x_{o_1}) \subseteq G_1(x_{o_1})$, $c(H(x_{o_1})) = \infty$. Finally, using Lemma 3 and 4, we can show that $\forall H'(x_{o_j}) \subseteq G_2(x_{o_j})$, where $x_{o_j} \in \mathcal{X}_{o_2}$, $c(H(x_{o_1}) \parallel H'(x_{o_j})) = \infty$, which means that there is no (DP-)optimal submachine of $G_1(x_{o_1}) \parallel G_2(x_{o_j}) = G_1 \parallel G_2(\langle x_{o_1}, x_{o_j}\rangle)$. ⋄

4.3.3 The optimal control of HFSMs: Next, we apply the previous results to the computation of the (DP)-optimal submachine of an HFSM. As in Section 4.2, we focus on a two levels HFSM. Let $\mathcal{K} = (\langle K_1, K_2, \ldots, K_n\rangle, (Y_1, I_1))$ be such an HFSM, then the DP-Optimal algorithm is given by:

step 1: $\forall i \geq 2$, compute the maximal DP-Optimal submachine H_i of K_i according to the algorithm DP-Opt [6].

step 2: $\forall b \in \mathcal{B}_1$, with $Y_1(b) = (K_j)_{j \in J_b}$ and $\forall x_o = \langle x_{o_1}, \ldots, x_{o_{\|J_b\|}}\rangle \in \prod_{j \in J_b} \mathcal{X}_{o_j}$,

1. We denote by $b_{|x_o}$ the FSM $\|_{j \in J_b} H_j(x_{o_j})$
2. $c(b_{|x_o}) = \sum_{j \in J_b} c(H_j(x_{o_j}))$ (note that if $x_{o_j} \notin \mathcal{X}_{H_{o_j}}$, we assume that $c(H_j(x_{o_j})) = \infty$)
3. $\forall \sigma \in I(b)(x_o)$, we rename σ as $\sigma_{b_{|x_o}}$ and we attach to this new event the cost $c_e(\sigma_{b_{|x_o}})$ defined by $c_e(\sigma_{b_{|x_o}}) = c_e(\sigma) + c(b_{|x_o})$[6] (the controllable status of $\sigma_{b_{|x_o}}$ is the same as σ'one).
   Call K'_1 the obtained structure.

step 3: Compute the DP-Optimal submachine of $K_1'^A$. Call H_1 the corresponding structure.

step 4: Call $\mathcal{K}_{opt} = (\langle H_1, \ldots., H_n\rangle, (Y'_1, I'_1))$, the resulting HFSM, where Y'_1, I'_1 are obtained by restricting Y_1 and I_1, according to the new transition relation of H_1, δ'_1.

Theorem 4 *$(\mathcal{K}_{opt})^F$ is a DP-Optimal submachine of $\mathcal{K}^F$.* ⋄

Following notations of Step 2, we first need the next lemma, for which we just give an intuitive proof:

Lemma 5 *Let $b \in \mathcal{B}_1$, and x_o an initial state of b. We note $b^{opt}_{|x_o}$ the FSM $\|_{j \in J_b} H_j(x_{o_j})$ and $\mathcal{K}^F[b_{|x_o} \to b^{opt}_{|x_o}]$, the FSM obtained by replacing $\mathcal{K}^F([b, x_o], [b, x_{f_b}])$ by $b^{opt}_{|x_o}$ in $\mathcal{K}^F$. Then $c((\mathcal{K}^F([b, x_o], [b, x_{f_b}])^{opt}) = c((\mathcal{K}^F)^{opt})$.* ⋄

The intuition behind this lemma is that we can replace a super-state by its corresponding optimal super-state in $\mathcal{K}$, without changing the global worst case cost of the DP-optimal submachine (i.e. the result of the DP-OPT algorithm of the resulting machine is a DP-Optimal submachine of $\mathcal{K}^F$). This is due to the fact that when entering a super-state b, the only way for the system to evolve out this super-state is its final state. Hence, in order to optimize the behavior of the system, we at least need to optimize the submachine that goes from one initial state to the final state of b. Proposition 6 tell us that $b^{opt}_{|x_o}$ is a good candidate for this optimization, since $b^{opt}_{|x_o}$ is DP-Optimal w.r.t. $\|_{j \in J_b} K_j(x_{o_j})$. Hence, it is sufficient, to first compute the costs of the super-states and then to consider them as atomic-state. To take into account their costs, we replace the cost of the events that lead to the super-states by adding to their initial cost, the corresponding cost of the super-states (computed in step 2(2)). This is exactly what we is done in Step 2(3). Further, it is sufficient to apply the DP-OPT algorithm on the new structure K'_1 seen as an FSM in order to obtain the result.

[6] Note that, due to Assumption 1, all the events of $\delta_1^{-1}(b)$ are taken into account

5 Conlusions and future works

In this paper, we introduced a class of DES modeled as Hierarchical Finite State machines, that can be seen as a simplified version of the STATECHARTS. Based on this model, we provided algorithms allowing the computation of non-blocking supervisors solving the forbidden state avoidance problem as well as the optimal control problem without expanding the HFSM. Moreover, the result is a collection of supervisors (one for each structure) that are generic enough to be computed only once and work in different contexts. We are currently working on control algorithms for more intricate control objectives (transversal forbidden state problem) as well as on the extension to a model with synchronized structures (i.e. synchronization on shared events).

References

[1] R. Alur and M. Yannakakis. Model checking of hierarchical state machines. In *Sixth ACM Symposium on the Foundations of Software Engineering*, volume 23, 6 of *Software Engineering Notes*, pages 175–188, New York, 1998.

[2] Y. Brave and M. Heimann. Control of discrete event systems modeled as hierarchical state machines. *IEEE Transacations on Automatic Control*, 38(12):1803–1819, December 1993.

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

[4] D. Harel and A. Pnueli. On the development of reactive systems. In *Logics and Models of Concurrent Systems*, volume 13 of *NATO ASI Series*, pages 477–498, New York, 1985.

[5] 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.

[6] R. Sengupta and S. Lafortune. An optimal control theory for discrete event systems. *SIAM Journal on Control and Optimization*, 36(2), March 1998.

[7] G. Shen and P. Caines. Control consistency and hierarchically accelerated dynamic programming. In 37*th IEEE Conference on Decision and Control*, pages 1686–1691, 1998.

[8] K. C. Wong and W. M. Wonham. Hierarchical control of discrete-event systems. *Discrete Event Dynamic Systems*, 6:241–273, 1996.