Verification of Succinct Hierarchical State Machines [⋆]

Salvatore La Torre[1], Margherita Napoli[1], Mimmo Parente[1], and Gennaro Parlato[1,2]

[1] Dipartimento di Informatica e Applicazioni
Università degli Studi di Salerno, Italy
[2] University of Illinois at Urbana-Champaign, USA

Abstract. A hierarchical state machine (HSM) is a finite state machine where a vertex can either expand to another hierarchical state machine (*box*) or be a basic vertex (*node*). Each node is labeled with atomic propositions. We study an extension of such model which allows atomic propositions to label also boxes (SHSM). We show that SHSMs can be exponentially more succinct than HSMs, and verification is in general harder by an exponential factor. Also, we show for a subclass of SHSMs (which can still be exponentially more succinct than HSMs) the same upper bounds as for HSMs.

1 Introduction

Finite state machines (labeled finite transition systems) are widely used for modeling the flow of control of digital systems and are appealing to formal verification such as model checking [CE81,CK96]. In model checking, a high-level specification is expressed by a formula of a logic and is checked for fulfillment on an abstract model of the system. Though a typical solution to this problem is linear in the size of the model, it is computationally hard since the model generally grows exponentially with the number of variables used to describe the system (*state-space explosion*). As a consequence, an important part of the research on model checking has been concerned with handling this problem.

Hierarchical structures are naturally present in complex systems where simple modules usually form a hierarchy. Representing such systems as standard transition graphs introduces some redundancy which can be avoided by modeling them as hierarchical state machines [AY01]. A *hierarchical finite state machine* is a finite state machine where a vertex can either expand to another hierarchical state machine (*box*) or be a basic vertex (*node*). Each node is labeled with atomic propositions (AP) and the outcomes of the model thus generate sequences over 2^{AP}. Complexity of model-checking is discussed in [AY01] and the succinctness of the model compared to standard finite state machines is addressed in [AKY99].

[⋆] This research was partially supported by the MIUR grant ex-60% 2005-06, Università degli Studi di Salerno.

In this paper, we consider a variation of the hierarchical state machines, where also boxes are labeled with atomic propositions. The intended meaning of such labeling is that when a box v expands to a machine M, all the vertices of M *inherit* the atomic propositions which hold true at v (*context*), so that different vertices expanding to M can place M into different contexts. With such more general labeling, we show that it is possible to obtain models of systems which are exponentially more succinct than hierarchical state machines. Therefore, we call them Succinct HSMs (SHSMs). We recall that a restricted version of an SHSM, denoted CHSM (Context-dependent HSM), is considered in [LNPP03]. There, an atomic proposition which labels a box b cannot label the vertices of any of the machines which directly or indirectly expand from b. We also prove that SHSMs can be exponentially more succinct than CHSMs.

We study the complexity of verification for SHSMs. In particular, we consider basic verification questions such as reachability and cycle detection, and the model-checking problem against LTL [Pnu77] and CTL [CE81] specifications. We show that for an SHSM $\mathcal{M}$ and a formula φ: LTL model-checking is PSPACE-complete and can be solved in $O(|\mathcal{M}|\,16^{|\varphi|})$ time; CTL model-checking is EXPTIME-complete and can be solved in $O(|\mathcal{M}|\,4^{|\varphi|\,d})$ time where d is the maximum number of exit nodes of $\mathcal{M}$. We also show that reachability and cycle detection are both NP-complete. From the results shown in [AY01] for hierarchical state machines, we get time complexities increased by an $O(2^{|\varphi|})$ factor which is exactly what we gain in succinctness. However, we also show that things improve when we place some restrictions on the SHSM and the formula expressing the target set in the reachability and cycle detection problems. In particular, we show that LTL model-checking of CHSMs can be solved in $O(|\mathcal{M}|\,8^{|\varphi|})$ time, and if the target formula conforms with the structure of the SHSM (resp. CHSM), reachability and cycle detection can be solved in linear time.

There are several papers in the literature that concern with hierarchical state machines. In [AGM00,AMY02], the verification tool HERMES which is based on hierarchical state machines is discussed. Hierarchical state machines with recursive expansions of nodes (recursive state machines) are studied in [ABE+05] and a corresponding temporal logic is introduced in [AEM04]. Recursive state machines turn out to be equivalent to pushdown automata [ABE+05]. Recursive calls and context-dependent properties are considered in [LNPP03], where also CHSMs are introduced as a restriction of the corresponding recursive model and some complexity results on verification are derived from those obtained on the recursive model. Here, we revisit such results and compare CHSMs to SHSMs. The impact of concurrency is studied in [AKY99,LMP04] for hierarchical state machines and in [BLP06] for recursive state machines. Finally, modular control synthesis for recursive state machines is studied in [ALM03a,ALM03b].

2 The Model

Given a set AP of atomic propositions, a *Kripke structure* over AP is a rooted directed graph whose vertices are labeled with the atomic propositions holding true in that vertex. In this paper, we model a system by a hierarchically structured graph, where vertices can be either simple nodes or placeholders for other graphs. We formally define such graphs as follows.

Definition 1. *A* Succinct Hierarchical State Machine *(SHSM) over AP is a tuple $\mathcal{M} = (M_1, \ldots, M_k)$, each $M_i = (N_i, in_i, \text{OUT}_i, \text{TRUE}_i, expn_i, E_i)$ is called* machine *and consists of:*

- *a finite set of vertices N_i, an* initial *vertex $in_i \in N_i$ and a set of* output *vertices $\text{OUT}_i \subseteq N_i$;*
- *a labeling function $\text{TRUE}_i : N_i \longrightarrow 2^{AP}$ that maps each vertex with a set of atomic propositions;*
- *an expansion mapping $expn_i : N_i \longrightarrow \{0, 1, \ldots, k\}$ such that $expn_i(u) < i$, for every $u \in N_i$, and $expn_i(u) = 0$, for every $u \in \{in_i\} \cup \text{OUT}_i$;*
- *a set of edges E_i where each edge is either a pair (u, v), with $u, v \in N_i$ and $expn_i(u) = 0$, or a triple $((u, z), v)$ with $u, v \in N_i$, $expn_i(u) > 0$, and $z \in \text{OUT}_{expn_i(u)}$.*

We assume that the sets of vertices N_i are pairwise disjoint. The set of all vertices of $\mathcal{M}$ is $N = \bigcup_{i=1}^{k} N_i$. The mappings $expn : N \longrightarrow \{0, 1, \ldots, k\}$ and $\text{TRUE} : N \longrightarrow 2^{AP}$ extend the mappings $expn_i$ and TRUE_i, respectively. If $expn(u) = j > 0$, the vertex u expands to the machine M_j and is called *box*. When $expn(u) = 0$, u is called a *node*. Let us define the closure $expn^+ : N \longrightarrow 2^{\{0,1,\ldots,k\}}$, as: $h \in expn^+(u)$ if either $h = expn(u)$ or there exists $u' \in N_{expn(u)}$ such that $h \in expn^+(u')$. We say that a vertex u is an *ancestor* of v, and v is a *descendant* from u, if $v \in N_h$ and $h \in expn^+(u)$. Let us note that contrarily with what happens in Kripke structures, in this model the atomic propositions not labeling u are not necessarily to be intended *false* (in Sect. 3 we define the function $\text{FALSE}(u)$ of the atomic proposition which can be stated *false* at u).

As an example of an SHSM $\mathcal{M}$ see part (a) of Fig. 1, where p_1, p_2, p_3 are atomic propositions labeling nodes and boxes of $\mathcal{M}$, e_i and x_i are respectively entry nodes and exit nodes for $i = 1, 2, 3$, and $expn(b_j^i) = j - 1$ for $i = 0, 1$ and $j = 2, 3$.

A *hierarchical state machine* (HSM) is an SHSM where $\text{TRUE}(u) = \emptyset$, for every box u [AY01]. A *context-dependent* HSM (CHSM) $\mathcal{M} = (M_1, \ldots, M_k)$ is an SHSM where $\text{TRUE}(u) \cap \text{TRUE}(v) = \emptyset$, for every $u, v \in N$ such that u is an ancestor of v [LNPP03]. Observe that the SHSM of Fig. 1 is also a CHSM but is not a HSM.

The semantics of an SHSM $\mathcal{M}$ is given by defining an equivalent Kripke structure, denoted $\mathcal{M}^F$.

The Kripke structure $\mathcal{M}^F$. A sequence of vertices $\alpha = u_i u_{i+1} \cdots u_j$, $0 < i \leq j$, is called a *well-formed sequence* if $u_{\ell+1} \in N_{expn(u_\ell)}$, for every $\ell = i, \ldots, j-1$. Moreover, α is also *complete* when $u_i \in N_k$ and u_j is a node.

A state of $\mathcal{M}^F$ is $\langle \alpha \rangle$ where α is a complete well-formed sequence of $\mathcal{M}$. Note that the length of a complete well-formed sequence is at most k, therefore the number of states of $\mathcal{M}^F$ is at most exponential in the number of vertices of $\mathcal{M}$. The initial state of $\mathcal{M}^F$ is $\langle in_k \rangle$, where in_k is the initial vertex of M_k.

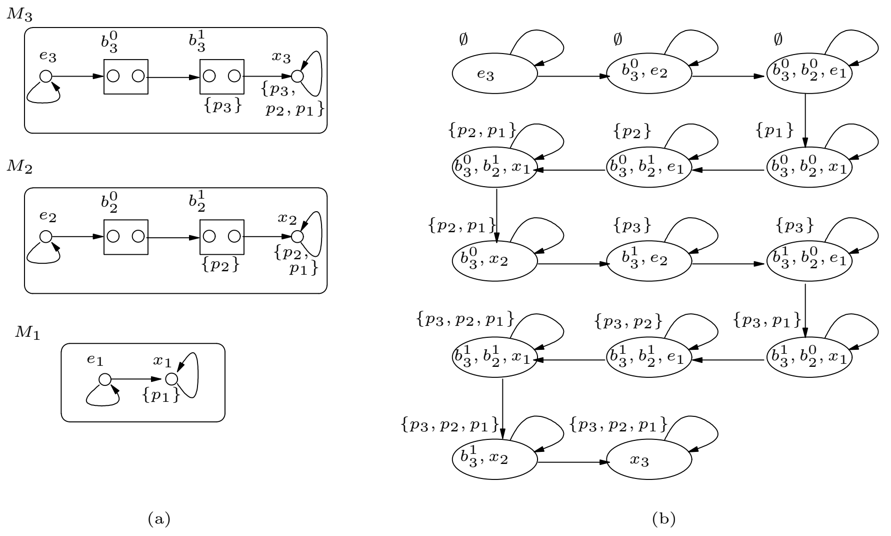

Fig. 1. An SHSM $\mathcal{M}$ and the Kripke structure obtained by flattening $\mathcal{M}$.

Transitions of $\mathcal{M}^F$ are obtained by using as templates the edges of $\mathcal{M}$: there is a transition from a state $\langle \alpha\beta \rangle$ to a state $\langle \alpha\beta' \rangle$ if and only if (β, β') "denotes" an edge of $\mathcal{M}$. Part (b) of Fig. 1 shows the Kripke structure which is equivalent to the SHSM from part (a) of the same figure. We give a precise construction of $\mathcal{M}^F$ in the following.

Given an SHSM $\mathcal{M} = (M_1, \ldots, M_k)$, note that the tuple $\mathcal{M}_h = (M_1, \ldots, M_h)$, $1 \leq h \leq k$, is an SHSM as well, and $\mathcal{M}_k = \mathcal{M}$. We now sketch how to compute recursively the flat Kripke structures $\mathcal{M}_h^F$, and thus $\mathcal{M}^F$.

We start with $\mathcal{M}_1^F$ which is obtained from machine M_1 by simply replacing each vertex u with a state $\langle u \rangle$ labeled with $\textsc{true}(\langle u \rangle) = \textsc{true}(u)$ (recall that by definition all vertices of $\mathcal{M}_1$ are nodes). Thus, for each edge $(v, w) \in E_1$ we add a transition $(\langle v \rangle, \langle w \rangle)$ in $\mathcal{M}_1^F$.
For $h > 1$, $\mathcal{M}_h^F$ is obtained from M_h by simply replacing each box u of M_h with a copy of the Kripke structure $\mathcal{M}_{expn(u)}^F$. More precisely, for each node $u \in N_h$, $\langle u \rangle$ is a state of $\mathcal{M}_h^F$ which is labeled with $\textsc{true}(u)$. For each box $u \in N_h$ and state $\langle \alpha \rangle$ of $\mathcal{M}_{expn(u)}^F$, $\langle u\alpha \rangle$ is a state of $\mathcal{M}_h^F$ and is labeled with $\textsc{true}(u) \cup \textsc{true}(\langle \alpha \rangle)$. The transitions of $\mathcal{M}_{expn(u)}^F$ are all inherited in $\mathcal{M}_h^F$, that is, there is a transition $(\langle u\alpha \rangle, \langle u\beta \rangle)$ in $\mathcal{M}_h^F$ for each transition $(\langle \alpha \rangle, \langle \beta \rangle)$ of $\mathcal{M}_{expn(u)}^F$. The remaining transitions of $\mathcal{M}_h^F$ correspond to the edges of M_h: for each node $v \in N_h$ and edge $(u, v) \in E_h$ (resp. $((u, z), v) \in E_h$) there is a transition from $\langle u \rangle$ (resp. $\langle uz \rangle$) to $\langle v \rangle$; for each box $v \in N_h$ and edge $(u, v) \in E_h$ (resp. $((u, z), v) \in E_h$) there is a transition from $\langle u \rangle$ (resp. $\langle uz \rangle$) to $\langle v\, in_{expn(v)} \rangle$.
A vertex u expanding into M_h is a placeholder for $\mathcal{M}_h^F$ and determines a subgraph in $\mathcal{M}^F$ isomorphic to $\mathcal{M}_h^F$. If two distinct vertices u_1 and u_2 both expand into the same machine M_h, that is $expn(u_1) = expn(u_2) = h$, then the states of

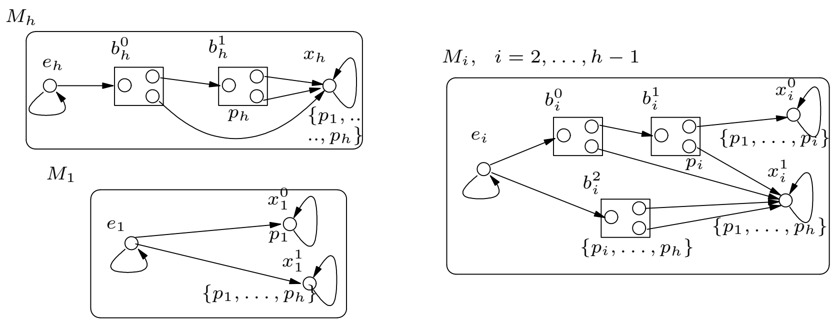

Fig. 2. An SHSM $\mathcal{M}_h$ such that $\mathcal{L}(\mathcal{M}_h, \varphi_h) = L_h^2$.

$\mathcal{M}_h^F$ appear in $\mathcal{M}^F$ in two different contexts, labeled possibly by different sets of atomic propositions: in one context this set contains TRUE(u_1) and in the other it contains TRUE(u_2).

Succinctness of the model. Observe that being able to represent with a single machine M_h more than one subgraph of $\mathcal{M}^F$, our model in general is more succinct than a traditional Kripke structure. Box labeling further improve on this. In fact in a situation as described above, the two 'copies' of $\mathcal{M}_h^F$ in $\mathcal{M}^F$, though labeled in a different way, can be represented in an SHSM by the single machine M_h, while they should be represented by two different machines in a HSM. Before stating this more formally, we need some definitions.

Given a transition graph with states labeled by subsets of atomic propositions and a state X, a *trace* is an infinite sequence $\sigma_1\sigma_2 \ldots \sigma_i \ldots$ of labels of states occurring in a path starting from X. Moreover, given an SHSM $\mathcal{M}$, we define the language $\mathcal{L}(\mathcal{M})$ as the set of the infinite traces of $\mathcal{M}^F$ starting from its initial state. Moreover, for a boolean formula φ over the atomic propositions AP, we denote by $\mathcal{L}(\mathcal{M}, \varphi)$ the set of traces in $\mathcal{L}(\mathcal{M})$ containing a label which fulfills φ. In the rest of this section, for $h \geq 0$ we fix a set of atomic propositions $\Sigma_h = \{p_1, \ldots, p_h\}$ and for a subset $\sigma \subseteq \Sigma_h$, define $\sharp(\sigma) = \sum_{p_i \in \sigma} 2^{i-1}$ (in particular, $\sharp(\emptyset) = 0$).

Proposition 1. *CHSMs can be exponentially more succinct than HSMs and finite state machines.*

Proof : Consider the family of languages L_h^1 of traces $\sigma_1 \ldots \sigma_n \ldots$ over 2^{Σ_h} such that: $\sigma_1 = \emptyset$; for $i < n$, $\sigma_i = \sigma_{i+1}$ or $\sharp(\sigma_{i+1}) = \sharp(\sigma_i) + 1$; and $\sharp(\sigma_n) = 2^h - 1$. Let $\varphi_h = p_1 \wedge \cdots \wedge p_h$. For $h = 3$, a CHSM $\mathcal{M}$ such that $\mathcal{L}(\mathcal{M}, \varphi_3) = L_3^1$ is given in Fig. 1. It is easy to see that $\mathcal{M}$ can be generalized to a CHSM $\mathcal{M}_h$ such that $\mathcal{L}(\mathcal{M}_h, \varphi_h) = L_h^1$, and $|\mathcal{M}_h| = O(h)$. Since there are 2^h different labels that need to be taken into account, we have that any hierarchical or finite state machine $\mathcal{M}'_h$ such that $\mathcal{L}(\mathcal{M}'_h, \varphi_h) = L_h^1$ requires at least 2^h different nodes. $\square$

There is an exponential gap also between CHSMs and SHSMs as shown in the following proposition.

Proposition 2. *SHSMs can be exponentially more succinct than CHSMs.*

Proof : Consider the family of languages L_h^2 of traces $\sigma_1 \ldots \sigma_n \ldots$ over 2^{Σ_h} such that: $\sigma_1 = \emptyset$; $\sharp(\sigma_n) = 2^h - 1$; and for $i < n$ either $\sigma_i = \sigma_{i+1}$, or $\sharp(\sigma_{i+1}) = \sharp(\sigma_i) + 1$, or for some $1 \leq j \leq h$, $\{p_1, p_2 \ldots p_{j-1}\} \subseteq \sigma_i$, $p_j \notin \sigma_i$ and $\sigma_{i+1} = \{p_j, \ldots, p_h\}$. Let $\varphi_h = p_1 \wedge \cdots \wedge p_h$. An SHSM $\mathcal{M}_h$ such that $\mathcal{L}(\mathcal{M}_h, \varphi_h) = L_h^2$ is given in Fig. 2, where $p_1, \ldots, p_h$ are atomic propositions labeling nodes and boxes of $\mathcal{M}$, e_i and x_i^j are respectively entry and exit nodes, and $expn(b_i^j) = i - 1$. It is easy to check that $|\mathcal{M}_h| = O(h)$.

To complete the proof it suffices to show that any CHSM $\mathcal{M}'_h$ such that $\mathcal{L}(\mathcal{M}'_h, \varphi_h) = L_h^2$ has size exponential in h. Fix $\sigma_1, \sigma_2 \subseteq \Sigma_h$ such that $p_j \notin \sigma_1 \cup \sigma_2$, $\{p_1, \ldots p_{j-1}\} \subseteq \sigma_1 \cap \sigma_2$ and there is an atomic proposition p_ℓ, $\ell > j$, such that $p_\ell \in \sigma_1$ and $p_\ell \notin \sigma_2$. Let $\mathcal{M}'_h$ be any CHSM such that $\mathcal{L}(\mathcal{M}'_h, \varphi_h) = L_h^2$. From the definition of L_h^2 and being $\sigma_1 \neq \sigma_2$, there are three different states s_1, s_2, s_3 of ${\mathcal{M}'_h}^F$ such that s_3 is a successor of s_2, $\textsc{true}(s_i) = \sigma_i$ for $i = 1, 2$ and $\textsc{true}(s_3) = \{p_j, \ldots, p_h\}$. For $i = 1, 2$, let $s_i = \langle \alpha_i b_i u_i \rangle$ where u_i are nodes, b_i are boxes, and α_i are sequences of boxes. Observe that from the definition of CHSM if p_ℓ labels any of the boxes in α_1 it cannot label any vertex of the machines to which either b_1 or u_1 belongs. Thus, either $b_1 \neq b_2$ or $u_1 \neq u_2$ must hold otherwise we get the contradiction that also s_3 (being a successor of s_2 in ${\mathcal{M}'_h}^F$) cannot be labeled with p_ℓ. Therefore, for each pair of different sets σ_1 and σ_2 as above, there are two different vertices of $\mathcal{M}'_h$. Since we can choose σ_1 and σ_2 among 2^{h-j} many different sets, we can conclude that any CHSM $\mathcal{M}'_h$ such that $\mathcal{L}(\mathcal{M}'_h, \varphi_h) = L_h^2$ must have at least 2^{h-j} different vertices. Therefore, if we pick $j = 1$ we get that such $\mathcal{M}'_h$ must have at least 2^{h-1} different vertices, and the proposition is proved. $\square$

It is worth noting that the mentioned succinctness results do not add up to each other, in the sense that it is not true that SHSMs can be doubly exponentially more succinct than HSMs. In fact, HSMs, CHSMs and SHSMs can all be translated to equivalent finite state machines with a single exponential blow-up. From Proposition 1 and the fact that any CHSM is also an SHSM, we have the following.

Corollary 1. *SHSMs can be exponentially more succinct than HSMs and finite state machines.*

3 Reachability and Cycle Detection

In this section we deal with reachability and cycle detection on SHSMs. We start defining the problems. Then, we show that they are NP-complete and discuss some conditions which allow us to give an efficient solution.

Given an SHSM $\mathcal{M} = (M_1, \ldots, M_k)$ and a propositional boolean formula φ, **Reach($\mathcal{M}$,φ)** is the problem of deciding if there exist a state X in $\mathcal{M}^F$ on which φ is satisfied and a path in $\mathcal{M}^F$ from $\langle in_k \rangle$ to X. The problem **Cycle($\mathcal{M}$,φ)** is the problem of deciding if there exist a state X in $\mathcal{M}^F$ at which φ is satisfied, a path from $\langle in_k \rangle$ to X and a path from X to itself.

NP-**completeness.** We first show NP-hardness. As stated in the following lemma, NP-hardness holds even when the formula is very simple.

Lemma 1. *Given an SHSM $\mathcal{M}$ and a formula φ expressed as a conjunction of literals, the problems* **Reach($\mathcal{M}$,φ)** *and* **Cycle($\mathcal{M}$,φ)** *are* NP*-hard.*

Proof : First, we observe that as the intuition suggests, **Cycle($\mathcal{M}$,φ)** can be polynomially reduced to **Reach($\mathcal{M}$,φ)**. Thus, to show the lemma we reduce 3-SAT to the problem **Reach($\mathcal{M}$,φ)**. Let ψ be a 3-SAT formula $\psi = C_1 \wedge \cdots \wedge C_n$, over a set of atomic propositions $AP' = \{P_1, \ldots, P_k\}$, where each C_i is a disjunction of three literals. We construct an SHSM $\mathcal{M} = (M_1, \ldots, M_k)$, on the set of atomic propositions $AP = \{c_1, \ldots, c_n\}$, as follows: each $M_i, i \geq 1$, has four vertices forming a line: in_i, p_i, $notp_i$ and out_i. The vertex p_i is labeled c_j, if P_i occurs in the clause C_j, while the vertex $notp_i$ is labeled c_j if $\neg P_i$ is in C_j. For $i > 1$, p_i and $notp_i$ are boxes expanding into M_{i-1}, having edges $((p_i, out_{i-1}), notp_i)$ and $((notp_i, out_{i-1}), out_i)$. Vertices p_1 and $notp_1$ are nodes. Define $\varphi = c_1 \wedge \cdots \wedge c_n$. It is not difficult to verify that ψ is satisfiable if and only if there exists a reachable state of $\mathcal{M}^F$ at which φ is satisfied. $\Box$

To show membership in NP, we introduce the notions of path and cycle on SHSMs. In the following, given a box $u \in N_h$, $\textsc{out}(u)$ is the set of all vertices $z \in \textsc{out}_{expn(u)}$ such that $((u, z), v) \in E_h$, for some $v \in N_h$. Moreover, with $[u, z]$ we denote a pair such that either u is a node and $z = \epsilon$ or u is a box and $z \in \textsc{out}(u)$.

Let $\mathcal{M} = (M_1, \ldots, M_k)$ be an SHSM. A *H-path* from a pair $[u_1, z_1]$ to a vertex u_r in M_h, is a sequence of pairs $[u_1, z_1]\,[u_2, z_2] \ldots [u_{r-1}, z_{r-1}]$ such that for every $1 \leq j < r$, $u_j \in N_h$ and one of the following cases occurs: either u_j is a node and $(u_j, u_{j+1}) \in E_h$, or u_j is a box, $((u_j, z_j), u_{j+1}) \in E_h$ and there is a H-path from $[in_{expn(u_j)}, \epsilon]$ to z_j in $M_{expn(u_j)}$. In the following, when we refer to a H-path in M_h, we sometimes omit the reference to a machine M_h since it is univocally determined by the vertices $u_1, \ldots, u_r$. Moreover, we say that a vertex $u \in N_h$ is *H-connected* if either $u = in_h$ or there is a H-path from $[in_h, \epsilon]$ to u. Observe that if a node $u \in N_h$ is H-connected then there exists a path from $\langle in_h \rangle$ to $\langle u \rangle$ in the Kripke structure $\mathcal{M}_h^F$, while if a box u is H-connected, a path exists from $\langle in_h \rangle$ to $\langle u\, in_{expn(u)} \rangle$. Hence, if a vertex u is H-connected, this in turn means that there exists a path from $\langle \alpha\, in_h \rangle$ to either $\langle \alpha u \rangle$ or $\langle \alpha u\, in_{expn(u)} \rangle$ in $\mathcal{M}^F$, for any well-formed sequence $\alpha = \epsilon$ or $\alpha = u_1 \ldots u_j$, with $expn(u_j) = h$. On the other side, note that if there exists a path in $\mathcal{M}^F$ from $\langle in_k \rangle$ to a state $\langle u_1 \ldots u_m \rangle$, then there are paths from $\langle u_1 \ldots u_{j-1}\, in_{expn(u_{j-1})} \rangle$ to $\langle u_1 \ldots u_{j-1} u_j\, in_{expn(u_j)} \rangle$ for $j = 1, \ldots, m-1$ and from $\langle u_1 \ldots u_{m-1}\, in_{expn(u_{m-1})} \rangle$ to $\langle u_1 \ldots u_{m-1} u_m \rangle$. Therefore, all vertices u_j for $j = 1, \ldots, m$ are H-connected. These observations prove the following proposition.

Proposition 3. *Given an SHSM $\mathcal{M} = (M_1, \ldots, M_k)$, let $X = \langle u_1 u_2 \ldots u_m \rangle$ be a state of $\mathcal{M}^F$. There is a path from $\langle in_k \rangle$ to X in $\mathcal{M}^F$ if and only if all the vertices u_i are H-connected.*

A *H-cycle* in an SHSM is a H-path from a pair $[u, z]$ to u. Now, we want to consider the relationship between cycles in $\mathcal{M}^F$ and H-cycles in $\mathcal{M}$. Given a pair $[v_0, z_0]$ with $z_0 \in \textsc{out}(v_0)$, a well-formed sequence $v_0 v_1 \ldots v_r$ is called *connected* to z_0 if for each $\ell = 1, \ldots, r$ there exist $z_\ell \in \textsc{out}(v_\ell) \cup \{\epsilon\}$, and a H-path from $[v_\ell, z_\ell]$ to $z_{\ell-1}$ (note that v_r may also be the output node z_{r-1}). This definition simply captures the property that there is a path of $\mathcal{M}^F$ from a state $\langle \alpha\, v_0 v_1 \ldots v_r \beta \rangle$ (with α, β possibly empty) to a state $\langle \alpha\, v_0 z \rangle$ such that $\alpha\, v_0$ is a common prefix to all the complete well-formed sequences of the visited states.

Proposition 4. *Let $\mathcal{M}$ be an SHSM and $X = \langle u_1 \ldots u_m \rangle$ be a reachable state of $\mathcal{M}^F$. X belongs to a cycle in $\mathcal{M}^F$ if and only if either:*

- *$[u_m, \epsilon]$ belongs to a H-cycle or*
- *there exist $1 \leq i < m$ and $z \in$OUT(u_i) such that $[u_i, z]$ belongs to a H-cycle and $u_i \ldots u_m$ is a well-formed sequence connected to z.*

The following proposition is crucial in our argument for NP-membership.

Proposition 5. *Given an SHSM $\mathcal{M}$, the set of H-connected vertices and the set of pairs belonging to a H-cycle of $\mathcal{M}$ can be determined in $O(|\mathcal{M}|)$.*

Therefore, we have the following theorem.

Theorem 1. *The problems* **Reach($\mathcal{M}$,φ)** *and* **Cycle($\mathcal{M}$,φ)** *for SHSMs are* NP*-complete.*

Efficient solutions. Here, we present some conditions on an SHSM $\mathcal{M}$ and a formula φ which allow us to give efficient algorithms for both **Reach($\mathcal{M}$,φ)** and **Cycle($\mathcal{M}$,φ)**. First we define a partial evaluation of φ on a well-formed sequence and then we give the algorithms for the above two problems.

Given a formula φ and two disjoint sets $T, F \subseteq AP$, let $\mathrm{Inst}(\varphi, T, F)$ denote the formula obtained by instantiating to *true* the atomic propositions of T and to *false* those in F. If we are given a state X of $\mathcal{M}^F$, then a propositional formula φ can be evaluated in X simply by computing $\mathrm{Inst}(\varphi, \textsc{true}(X), AP \setminus \textsc{true}(X))$. To solve our problems, we would like to find a state in $\mathcal{M}^F$ where φ is satisfied, using the SHSM $\mathcal{M}$, without explicitly constructing $\mathcal{M}^F$. To this aim we use a greedy approach to evaluate φ: we visit top-down $\mathcal{M}$ and at each vertex u we instantiate as many atomic propositions as possible. The question is: which are the atomic propositions of φ can be instantiated? Surely, we can instantiate to *true* all the atomic propositions of $\textsc{true}(u)$, while the atomic propositions which can be instantiated to *false* depend on the vertices that may follow u in any complete well-formed sequence of $\mathcal{M}$. In other terms, an atomic proposition can be instantiated to *false* if it labels neither u nor vertices having u as an ancestor, that is it does not belong to a set $\textsc{true}^*(u)$ defined as $\textsc{true}^*(u) = \textsc{true}(u) \cup \bigcup_{v \in N_{expn^+(u)}} \textsc{true}(v)$. Thus, we define the set of atomic propositions of φ that can be instantiated to *false* at a vertex $u \in N_h$, as $\textsc{false}(u) = AP \setminus \textsc{true}^*(u)$. Let us remark that the function $\textsc{false}$ can be computed by visiting $\mathcal{M}$ just once.

Given an SHSM $\mathcal{M}$, a propositional boolean formula φ over AP and a well-formed sequence α of $\mathcal{M}$, the *partial evaluation* of φ on α, is defined as

$$\begin{cases} \mathrm{PEval}(\varphi, \epsilon) = \varphi \\ \mathrm{PEval}(\varphi, \alpha u) = \mathrm{Inst}(\mathrm{PEval}(\varphi, \alpha), \textsc{true}(u), \textsc{false}(u)). \end{cases}$$

The following lemma ensures correctness of our approach.

Lemma 2. *Given an SHSM $\mathcal{M}$, a formula φ, and a state $X = \langle\alpha\rangle$ of $\mathcal{M}^F$: $PEval(\varphi, \alpha) = Inst(\varphi, \text{TRUE}(X), AP \setminus \text{TRUE}(X))$.*

In what follows w.l.o.g. we assume that the formula returned by a partial evaluation is simplified according to the following tautologies: $(\psi\wedge$ *true*$) \equiv \psi$, $(\psi\wedge$ *false*$) \equiv$ *false*, $(\psi\vee$ *false*$) \equiv \psi$, $(\psi\vee$*true*$) \equiv$ *true*, $(\neg$*true*$) \equiv$ *false*, $(\neg$ *false*$) \equiv$ *true*. We say that φ is *constant* if it is either *true* or *false*.
Now, we are ready to define a condition to get polynomial-time algorithms for the problems **Reach(**$\mathcal{M}$,φ**)** and **Cycle(**$\mathcal{M}$,φ**)**.

Fix an SHSM $\mathcal{M}$ and a boolean formula φ. We say that the partial evaluations of φ in $\mathcal{M}$ are *uniquely inherited* iff: for every two well-formed sequences αu and βv, if $expn(u) = expn(v)$ and both PEval$(\varphi, \alpha u)$ and PEval$(\varphi, \beta v)$ are not constant, then PEval$(\varphi, \alpha u)$ = PEval$(\varphi, \beta v)$.

Assume w.l.o.g. that all the vertices of $\mathcal{M}$ are H-connected (from Proposition 5, an arbitrary SHSM can be simplified such that this holds using linear time). Thus from Proposition 3 all states of $\mathcal{M}^F$ are reachable. Hence, using mainly Lemma 2, it is possible to design an algorithm that looks for a complete well-formed sequence α for which the function PEval(φ, α) gives TRUE. Such algorithm visits top-down the machines of $\mathcal{M}$ and evaluates PEval on well-formed sequences by computing iteratively the function Inst. Note that if the evaluations of φ are uniquely inherited, multiple visits of a machine M_h are not needed, and thus the overall complexity of the algorithm is linear in $|\mathcal{M}|$ and $|\varphi|$.

Theorem 2. *Let φ be a formula and $\mathcal{M}$ be an SHSM such that the partial evaluations of φ in $\mathcal{M}$ are uniquely inherited. The problem* **Reach(**$\mathcal{M}$,φ**)** *is decidable in time $O(|\mathcal{M}| \cdot |\varphi|)$.*

The algorithm to solve the problem **Cycle(**$\mathcal{M}$,φ**)** and its correctness strongly rely on the following Lemma 3 which is a direct consequence of Proposition 4.

Lemma 3. *Let $\mathcal{M}$ be an SHSM with all H-connected vertices, φ be a boolean formula and $\alpha = u_1 \dots u_j$ be a well-formed sequence such that $PEval(\varphi, \alpha)$ =*true. *There exists a well-formed sequence β such that $\langle\alpha\beta\rangle$ is a solution of the problem* **Cycle(**$\mathcal{M}$,φ**)** *if and only if either*

a) there exists a H-cycle in M_h, $h \in expn^+(u_j)$ or
b) there exist $i \leq j$ and $z_0 \in \text{OUT}(u_i)$ such that $[u_i, z_0]$ is in a H-cycle and $u_i \dots u_j$ is a well-formed sequence connected to z_0.

Theorem 3. *Let φ be a formula and $\mathcal{M}$ be an SHSM such that the partial evaluations of φ in $\mathcal{M}$ are uniquely inherited. The problem* **Cycle(**$\mathcal{M}$,φ**)** *is decidable in time $O(|\mathcal{M}| \cdot |\varphi|)$.*

Note that for general φ and $\mathcal{M}$, the above algorithms can be easily modified in such a way that each machine M_h can be visited just once for each subformula of φ. Thus the following theorem holds.

Theorem 4. *Given an SHSM $\mathcal{M}$ and a formula φ, the problems* **Reach(**$\mathcal{M}$,φ**)** *and* **Cycle(**$\mathcal{M}$,φ**)** *are decidable in $O(|\mathcal{M}| \cdot 2^{|\varphi|})$ time.*

Observe that when in $\mathcal{M}$ there are no context-properties ($\text{TRUE}(u) = \emptyset$ for every box u), then the partial evaluations of every formula φ in $\mathcal{M}$ are uniquely inherited and thus Theorems 2 and 3 generalize those in [AY01].

We give a further condition on $\mathcal{M}$ and φ which let us get efficient algorithms when $\mathcal{M}$ is a CHSM. Let $\text{AP}(\psi)$ be the set of atomic propositions of a formula ψ. We say that ψ is *local* to $\mathcal{M}$ vertices iff for every vertex u of $\mathcal{M}$ either $AP(\varphi_i) \cap (\text{TRUE}(u) \cup \text{FALSE}(u)) = \emptyset$ or $AP(\varphi_i) \cap (\text{TRUE}(u) \cup \text{FALSE}(u)) = AP(\varphi_i)$ holds. It is possible to show that for a CHSM $\mathcal{M}$, if φ is of the form $\varphi_1 \wedge \cdots \wedge \varphi_m$ where each φ_i is local to $\mathcal{M}$ vertices, then the partial evaluations of φ in $\mathcal{M}$ are uniquely inherited.

Corollary 2. *Let $\mathcal{M}$ be a CHSM and φ be a conjunction of formulas which are local to $\mathcal{M}$ vertices. The problems* **Reach($\mathcal{M}$,φ)** *and* **Cycle($\mathcal{M}$,φ)** *are decidable in time $O(|\mathcal{M}| \cdot |\varphi|)$.*

As a final remark, note that from Lemma 1, **Reach($\mathcal{M}$,φ)** and **Cycle($\mathcal{M}$,φ)** are NP-hard for SHSMs even if we restrict to conjunctions of formulas which are local to the vertices of the considered SHSM.

4 Ltl and Ctl Model Checking

In this section we consider the verification of requirements expressed by Ltl and Ctl formulas. We recall a Büchi automaton $A = (Q, q_1, \Delta, L, T)$ is a Kripke structure (Q, q_1, Δ, L) together with a set of accepting states T [Tho90].

Fix an SHSM $\mathcal{M} = (M_1, \ldots, M_k)$ and a Büchi automaton $A = (Q, q_1, \Delta, L, T)$, with $Q = \{q_1, \ldots, q_m\}$. We define the graphs $M_{(i,j,P)}$, where $1 \le i \le k$, $1 \le j \le m$, and P is a subset of AP such that $P \cup \text{TRUE}_{\mathcal{M}}(in_i) = L(q_j)$. The vertices of $M_{(i,j,P)}$ are 4-tuples $[u, q, j, P]$. The third and fourth components are the same for all the vertices in the same graph and are needed only to distinguish vertices of different graphs. Components u and q are respectively vertices of M_i and A such that: if u is a node, the labeling of q coincides with the labeling of u augmented with the set P of the atomic propositions that u inherits from its ancestors; if u is a box, with $expn(u) = h$, the labeling of q contains also the atomic propositions labeling the initial node in_h of the expansion of u. Moreover, the edges in $M_{(i,j,P)}$ are obtained from the edges of M_i and A as in the standard Cartesian product of M_i and A.

The SHSM $\mathcal{M}' = \mathcal{M} \bigotimes A$ is inductively defined as follows. $M_{(k,1,\emptyset)}$ is the graph containing the starting node of $\mathcal{M}'$. Let $M_{(i,j,P)}$ be a graph of $\mathcal{M}'$, and $[u, q_t, j, P]$ be a vertex of $M_{(i,j,P)}$. If $expn_{\mathcal{M}}(u) = 0$ then $expn_{\mathcal{M}'}([u, q_t, j, P]) = 0$. If $expn_{\mathcal{M}}(u) = h > 0$, and $P' = P \cup \text{TRUE}_{\mathcal{M}}(u)$ then $M_{(h,t,P')}$ is a graph of $\mathcal{M}'$ and $expn_{\mathcal{M}'}([u, q_t, j, P])$ is the index of $M_{(h,t,P')}$. Finally, we use only an atomic proposition TGT to label vertices: $\text{TRUE}_{\mathcal{M}'}([u, q, j, P])$ is $\{\text{TGT}\}$ iff $q \in T$.
By a counting argument and the observation that for CHSMs the graphs $M_{(i,j,P)}$ are uniquely determined by the indices i and j, we can show the following lemma.

Lemma 4. *Given an SHSM $\mathcal{M}$, $\mathcal{M}' = \mathcal{M} \bigotimes A$ is an SHSM whose size is $O(m^2 \cdot |\mathcal{M}| \cdot |A| \cdot |2^{AP}|)$. Moreover, if $\mathcal{M}$ is a CHSM, then the size of $\mathcal{M}'$ is $O(m^2 \cdot |\mathcal{M}| \cdot |A|)$.*

The language $\mathcal{L}(A)$ accepted by a Büchi automaton A is the set of all traces corresponding to paths visiting infinitely often a state of T. The SHSM $\mathcal{M} \bigotimes A$ can be used to check for emptiness the language given by the intersection of $\mathcal{L}(\mathcal{M})$ and $\mathcal{L}(A)$, as shown in the following lemma.

Lemma 5. *Emptiness of $\mathcal{L}(\mathcal{M}) \cap \mathcal{L}(A)$ can be checked in time linear in the size of $\mathcal{M} \bigotimes A$.*

Proof : Observe that the set of the traces of ${\mathcal{M}'}^F$ is the same as the set of traces of the Cartesian product of $\mathcal{M}^F$ and A. Thus $\mathcal{L}(\mathcal{M}) \cap \mathcal{L}(A) \neq \emptyset$ if and only if **Cycle($\mathcal{M}'$,**TGT**)** is TRUE. Since the simple formula consisting of the sole atomic proposition TGT has only a partial evaluation, that is TGT itself, from Theorem 3 this problem can be checked in linear time. □

As a consequence of the above lemmas, we obtain an algorithm to solving the LTL model checking for SHSMs. We construct a Büchi automaton $A_{\neg\varphi}$ of size $O(2^{|\varphi|})$ accepting the set $\mathcal{L}(A_{\neg\varphi})$ of the sequences that do not satisfy φ, and then φ is satisfied on all paths of $\mathcal{M}$ if and only if $\mathcal{L}(\mathcal{M}) \cap \mathcal{L}(A_{\neg\varphi})$ is empty. Therefore, by Lemmas 4 and 5 we have:

Theorem 5. *The* LTL *model checking on an SHSM $\mathcal{M}$ and a formula φ can be solved in $O(|\mathcal{M}| \cdot 16^{|\varphi|})$ time. Moreover, if M is a CHSM then the problem can be solved in $O(|\mathcal{M}| \cdot 8^{|\varphi|})$ time.*

Now we consider CTL for expressing branching-time properties and sketch an algorithm to solve CTL model checking. We fix an SHSM $\mathcal{M} = (M_1, \ldots, M_k)$ and a temporal logic formula φ. Let P_φ be the set of atomic propositions that appear in φ. The first step of our algorithm consists of constructing a hierarchical state machine $\mathcal{M}_\varphi$ such that $\mathcal{M}_\varphi^F$ is isomorphic to $\mathcal{M}^F$. Let $\mathit{index} : [1,k] \times 2^{P_\varphi} \rightarrow [1, k\, 2^{|P_\varphi|}]$ be a bijection such that $\mathit{index}(i, P_i) < \mathit{index}(j, P_j)$ whenever $i < j$. Clearly, index maps the pairs (i, P_i) into a strictly increasing sequence of consecutive naturals starting from 1. For a machine $M_i = (N_i, in_i, OUT_i, \textsc{true}_i, expn_i, E_i)$, $1 \leq i \leq k$ and $P \subseteq P_\varphi$, define M_i^P as the machine $(N_i^P, in_i^P, OUT_i^P, \textsc{true}_i^P, expn_i^P, E_i^P)$ where:

- $N_i^P = \{u^P \mid u \in N_i\}$, and $OUT_i^P = \{u^P \mid u \in OUT_i\}$;
- $\textsc{true}_i^P(u^P) = \textsc{true}_i(u) \cup P$ if u is a node and $\textsc{true}_i^P(u^P) = \emptyset$, otherwise;
- $expn_i^P(u) = 0$ if u is a node and $expn_i^P(u) = \mathit{index}(expn_i(u), P \cup \textsc{true}_i(u))$, otherwise;
- $E_i^P = \{(u^P, v^P) \mid (u,v) \in E_i\} \cup \{((u^P, z^{P \cup \textsc{true}_i(u)}), v^P) \mid ((u,z), v) \in E_i\}$.

We define $\mathcal{M}_\varphi$ by the tuple of machines $(M'_1, \ldots, M'_{k'})$ such that for $j = 1, \ldots, k'$, $M'_j = M_i^P$ where $j = \mathit{index}(i, P)$. From the definition of M_i^P it is simple to verify that $\mathcal{M}_\varphi$ is a HSM and $|\mathcal{M}_\varphi|$ is $O(|\mathcal{M}|\, 2^{|P_\varphi|}) = O(|\mathcal{M}|\, 2^{|\varphi|})$. Moreover, $\mathcal{M}_\varphi^F$ and $\mathcal{M}^F$ are identical up to a renaming of states. Therefore, by the results on HSMs from [AY01], we get the following theorem (where $\mathcal{M}$ is an SHSM, φ is a formula and d is the maximum number of exit nodes of a machine of $\mathcal{M}$).

Theorem 6.
The CTL *model checking of SHSMs can be solved in $O(|\mathcal{M}|\, 4^{|\varphi|\, d})$ time.*

References

[ABE+05] R. Alur, M. Benedikt, K. Etessami, P. Godefroid, T. Reps, and M. Yannakakis. Analysis of recursive state machines. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 27(4): 786–818, 2005.

[AEM04] R. Alur, K. Etessami, and P. Madhusudan. A Temporal Logic of Nested Calls and Returns. In *Proc. 10th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, TACAS'04*, LNCS 2988, pp. 467–481, 2004.

[AGM00] R. Alur, R. Grosu, and M. McDougall. Efficient reachability analysis of hierarchical reactive machines. In *Proc. 12th International Conference on Computer Aided Verification, CAV'00*, LNCS 1855, pages 280–295. Springer, 2000.

[AKY99] R. Alur, S. Kannan, and M. Yannakakis. Communicating hierarchical state machines. In *Proc. 26th International Conference on Automata, Languages and Programming, ICALP'99*, LNCS 1644, pp. 169–178, 1999.

[ALM03a] R. Alur, S. La Torre, and P. Madhusudan. Modular Strategies for Recursive Game Graphs. In *Proc. 9th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, TACAS'03*, LNCS 2619, pp. 363-378, 2003.

[ALM03b] R. Alur, S. La Torre, and P. Madhusudan. Modular Strategies for Infinite Games on Recursive Graphs. In *Proc. 15th International Conference on Computer Aided Verification, CAV'03*, LNCS 2725, pp. 67-79, 2003.

[AMY02] R. Alur, M. McDougall, and Z. Yang. Exploiting Behavioral Hierarchy for Efficient Model Checking. In *Proc. 14th International Conference on Computer Aided Verification, CAV'02*, LNCS 2404, pp. 338–342, 2002.

[AY01] R. Alur and M. Yannakakis. Model checking of hierarchical state machines. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 23(3):273 – 303, 2001.

[BLP06] L. Bozzelli, S. La Torre, and A. Peron. Verification of well-formed Communicating Recursive State Machines. In *Proc. 7th International Conference on Verification, Model Checking, and Abstract Interpretation, VMCAI'06*, LNCS 3855, pp. 412–426, 2006.

[CE81] E.M. Clarke and E.A. Emerson. Design and synthesis of synchronization skeletons using branching time temporal logic. In *Proc. of Workshop on Logic of Programs*, LNCS 131, pages 52 – 71. Springer-Verlag, 1981.

[CK96] E.M. Clarke and R.P. Kurshan. Computer-aided verification. *IEEE Spectrum*, 33(6):61 – 67, 1996.

[LNPP03] S. La Torre, M. Napoli, M. Parente, G. Parlato. Hierarchical and Recursive State Machines with Context-Dependent Properties. In *Proc. 30th International Conference on Automata, Languages and Programming, ICALP'03*, LNCS 2719, pp. 776–789, 2003.

[LMP04] R. Lanotte, A. Maggiolo, and A. Peron, Structural Model Checking for Communicating Hierarchical Machines. In *Proc. 31st International Symposium on Mathematical Foundations of Computer Science, MFCS'04*, LNCS 3153, pp. 525–536, 2004.

[Pnu77] A. Pnueli. The temporal logic of programs. In *Proc. 18th IEEE Symposium on Foundations of Computer Science, FOCS'77*, pages 46 – 77, 1977.

[Tho90] W. Thomas. Automata on infinite objects. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 133 – 191. Elsevier Science Publishers, 1990.