Complexity Aspects of Web Services Composition

Karima Ennaoui, Lhouari Nourine, and Farouk Toumani

LIMOS, CNRS, Universite Clermont Auvergne
1 Rue de la Chebarde, 63178 AUBIERE CEDEX

Abstract. The web service composition problem can be stated as follows: given a finite state machine M, representing a service business protocol, and a set of finite state machines $\mathcal{R}$, representing the business protocols of existing services, the question is to check whether there is a simulation relation between M and the shuffle product closure of $\mathcal{R}$. In fact the shuffle product is a subclass of the communication free petri net and basic parallel processes, for which the same problem of simulation is known to be 2-Exptime-hard.
This paper studies the impact of several parameters on the complexity of this problem. We show that the problem is *Exptime-complete* if we bound either: *(i)* the number of instances of services in $\mathcal{R}$ that can be used in a composition, or *(ii)* the number of instances of services in $\mathcal{R}$ that can be used in parallel, or *(iii)* the number of the so-called hybrid states in the finite state machines of $\mathcal{R}$ by 2.

1 Introduction

Web Services [2] is a new computing paradigm that tends to become a technology of choice to facilitate interoperation among autonomous and distributed applications. The UDDI consortium defines Web services as *self-contained, modular business applications that have open, Internet-oriented, standards-based interfaces.* Several models have been proposed in the literature to describe different facets of services. In particular, the importance of specifying external behaviour of services, also called service business protocols, has been highlighted in several research works [6,3,4]. Through literature, different models have been used to represent web service business protocols. The Finite States Machines (FSM) formalism is widely adopted in this context to model *statefull* applications exposed as web services where states represent the different phases that a service may go through while transitions represent "abstract" activities that a service can perform [6,3,4].

We consider in this paper the problem of Web Service Composition (WSC). This problem arises from the situation where none of the existing services can provide a requested functionality. In this case, the idea is to find out, algorithmically, if the target functionality could be composed out of the existing services (components repository). This automatic approach of composition simplifies the development of software by reusing existing components and offers capabilities to customize complex systems built on the fly [9]. We focus more particularly on a specific instance of WSC, namely the (business) protocol synthesis problem, which can be stated as follows: given a set of business protocols of available services and given a business protocol of a target service, is it possible to synthesize automatically a mediator that *implements* the target service using the existing ones?

[16] shows that when business protocols are described by means of FSMs, the WSC problem can then be formalized as the problem of deciding whether there exists a simulation relation between the target protocol and the shuffle (or asynchronous product) of the available ones. This result is however based on the implicit assumption that at most one *instance* of each available service can be used in a composition. This setting has been extended in [9] to the case where the number of instances that can be used in a composition is unbounded. WSC is formalized in this latter case as a simulation problem between an FSM and an infinite state machine, called Product Closure State Machine (PCSM), that is able to compute the shuffle closure of an FSM.

Shuffle product of FSMs (and PCSM) is a subclass of Basic Parallel Processes (BPP), the class of communication free petri nets: every transition has at most one input place. Simulation of FSM by BPP was proven Expspace-hard by Lasota [15] and 2-Exptime-hard in [8].

Complexity analysis of WSC was first considered by Musholl *et al.*[16], under the aforementioned implicit assumption, where it is shown Exptime-Complete. In case of unbounded instances, the WSC problem has been proved decidable with an Ackermanian function as upper bound in [9]. The proof of [9] is based on Dickson lemma, and hence cannot be exploited to derive tighter upper bounds. An Expspace-hard lower bound is given by Lasota[15]. The source of complexity derived from the analysis of the algorithm given in [9] is related to the presence of the so-called hybrid states (final states with outgoing transitions and correspond to unbounded places in Petri net terminology) in the components and loops in the target: if the target FSM is loop free, the WSC problem becomes NP-complete and when the components are hybrid state free the problem is proven Exptime.

In this paper, we consider additional parameters related to bounded/unbounded web services composition. We consider as inputs an FSM M (the target protocol) and a set of FSMs $\mathcal{R}$ (the protocols of the available services) and we investigate the complexity of testing simulation between M and the shuffle closure of $\mathcal{R}$, represented as a PCSM [9]. More precisely, we study the complexity of the following problems:

1. $WSC(M, \mathcal{R})$: The problem of composing M using an unbounded number of instances of $\mathcal{R}$.
2. $BC(M, \mathcal{R}, k)$: The problem of composing M using at most k instances of each FSM in $\mathcal{R}$.
3. $PBC(M, \mathcal{R}, k)$: The problem of composing M using simultaneously at most k FSM instances in $\mathcal{R}$ (in parallel).
4. $UCHS(M, \mathcal{R}, k)$: The problem of composing M using an unbounded number of instances of $\mathcal{R}$, with the number of hybrid states in $\mathcal{R}$ is bounded by $k \in \{0, 1, 2\}$.

Table in figure 1 displays known and new complexity results regarding the WSC problem.

M	Acyclic FSM	general FSM
$BC(M, \mathcal{R}, 1)$	NP-complete[9]	Exptime-complete [16]
$BC(M, \mathcal{R}, k)$	NP-complete[9]	Exptime-complete
$PBC(M, \mathcal{R}, 1)$	Polynomial	Polynomial
$PBC(M, \mathcal{R}, k)$	NP-complete	Exptime-complete
$WSC(M, \mathcal{R})$	NP-complete [9]	Decidable [9]
$UCHS(M, \mathcal{R}, 0)$	NP-complete[9]	Exptime-complete
$UCHS(M, \mathcal{R}, 1)$	NP-complete[9]	Exptime-complete
$UCHS(M, \mathcal{R}, 2)$	NP-complete[9]	Exptime-complete

Fig. 1. Complexity results of WSC sub-problems

Paper organisation Section 2 recalls some basic definitions needed in this paper. In section 3, we investigate the problem of bounded web services composition and proves that it is Exptime-Complete. Next, we define web service composition with fixed number of parallel instances, and show that it is Exptime-Complete in general and is NP-complete when M is loop free and polynomial for $k = 1$. In section 5, we consider the web service composition when the number of hybrid states is bounded. We show that this problem is Exptime-Complete for $k = 0$, $k = 1$ and $k = 2$. We conclude in section 6.

2 Preliminaries

Finite State Machine We consider in this paper service business protocols formally described as FSMs. We recall below the definition of such machines.

Definition 1. ***(Finite State Machine (FSM))***
A State Machine ***(SM)*** *M is a tuple $M = (\Sigma_M, Q_M, F_M, q_M^0, \delta_M)$, where: Σ_M is a finite alphabet, Q_M is a set of states, $\delta_M \subseteq Q_M \times \Sigma_M \times Q_M$ is a set of labelled transitions, $F_M \subseteq Q_M$ is a set of final states, and $q_M^0 \in Q_M$ is the initial state. If Q_M is finite then M is called a Finite State Machine (FSM).*

Moreover, a state $q \in Q_M$ is called: **intermediate**, if $q \notin F_M$ and $\exists p_1, p_2 \in Q_M$, $s.t$ $(p_1, a, q) \in \delta_M$ and $(q, b, p_2) \in \delta_M$, we denote by $I(M)$ the set of intermediate states of M; **hybrid**, if $q \in F_M$, $q \neq q_0$ and there exist at least one transition $(q, b, p) \in \delta_M$, with $p \in Q_M$ and $b \in \Sigma$, the set of hybrid states is denoted $H(M)$ and **terminal**, if $q \in F_M$ and is not hybrid.

We define the **norm of a state** q as the finite length of the shortest path from q to a final state. **The norm of an FSM** M, noted *norm(M)*, is the maximal norm of its states.

k-Iterated Product Machine (k-IPM) and Product State Machine (PCSM) We start by defining the shuffle (asynchronous product) and union operations on FSMs:

Definition 2. ***(Asynchronous product and Union of two FSMs)*** *Let $M = (\Sigma_M, Q_M, F_M, q^0_M, \delta_M)$ and $M' = (\Sigma_{M'}, Q_{M'}, F_{M'}, q^0_{M'}, \delta_{M'})$ be two FSMs. We have :*

- *The* ***shuffle or asynchronous product*** *of M and M', denoted $M \times M'$, is an FSM $(\Sigma_M \cup \Sigma_{M'}, Q_M \times Q_{M'}, F_M \times F_{M'}, (q^0_M, q^0_{M'}), \lambda)$ where the transition function λ is defined as follows: $\lambda = \{((q, q'), a, (q_1, q_1')) : ((q, a, q_1) \in \delta_M$ and $q' = q'_1)$ or $((q', a, q_1') \in \delta_{M'}$ and $q = q_1)\}$.*
- *The* ***union*** *of M and M', denoted $M \cup M'$, is the FSM $(\Sigma_M \cup \Sigma_{M'} \cup \{\epsilon\}$, $Q_M \cup Q_{M'} \cup \{q_0\}$, $F_M \cup F_{M'}$, q_0, $\delta_M \cup \delta_{M'} \cup \{(q_0, \epsilon, q^0_M), (q_0, \epsilon, q^0_{M'})\})$.*

For a set of available FSMs $\mathcal{R} = \{M_1, ...M_i\}$, we consider a compact structure that abstracts all possible executions that can be produced using the components of $\mathcal{R}$. First, we begin by the simple case where each M_j can be used only once:

Definition 3. ***(Union of asynchronous products of FSMs set)*** *Let $\mathcal{R} = \{M_1....M_m\}$ be a FSMs repository. We define $\odot(\mathcal{R})$ the union of asynchronous product of all the subsets of $\mathcal{R}$ as the FSM: $\odot(\mathcal{R}) = \bigcup_{\{M_{i_1},...,M_{i_j}\} \in 2^{\mathcal{R}}} (M_{i_1} \times ... \times M_{i_j})$ where $j \in [0, i]$.*

Second, we consider the case where the number of copies of each $M_j \in \mathcal{R}$ is bounded by an integer k:

Definition 4. ***(k-iterated product of FSMs set $\mathcal{R}$)*** *The* ***k-iterated product of*** *$\mathcal{R}$ is defined by $\mathcal{R}^{\otimes_k} = \mathcal{R}^{\otimes_{k-1}} \times \odot(\mathcal{R})$ with $\mathcal{R}^{\otimes_1} = \odot(\mathcal{R})$.*

Finally, we consider the general case where the number of instances of each $M_j \in \mathcal{R}$ is unbounded. This corresponds to the product closure of $\mathcal{R}$ [9]:

Definition 5. ***(Product closure of FSMs set)*** *The* ***product closure*** *of $\mathcal{R}$, noted $\mathcal{R}^{\otimes}$, is defined as: $\mathcal{R}^{\otimes} = \bigcup_{i=0}^{+\infty} \mathcal{R}^{\otimes_i}$.*

The **P**roduct **C**losure **M**achine (**PCSM**) of $\mathcal{R}$, defined in [9] and proven equivalent to $\mathcal{R}^{\otimes}$, is the SM with unbounded number of tokens stacked at the beginning in the initial states in $\mathcal{R}$. Then, the instantaneous description of a PCSM gives the number of tokens (instances) at each state of its underlaying FSMs. This description is called a configuration of $\mathcal{R}^{\otimes}$. We omit from this description the initial states (source:infinite number of tokens) and terminal states (sink:terminated instances).

Example 1. Figure 2 illustrates the execution of the sequence "*abca*" by the PCSM of the FSM M in figure 2-(a). M contains one intermediate state q_1 and two hybrid states q_2 and q_5. Therefore, figure 2-(b) depicts a part of the PCSM $M^{\otimes}$ with triplets as configurations where integers witness respectively the number of tokens in q_1, q_2 and q_5. For each configuration c in figure 2-(b), we associate an instant t (or several instants) during the execution when c describes the PCSM. At the beginning ($t = 0$), $M^{\otimes}$'s instantaneous description is $(0, 0, 0)$, interpreting an empty stack in every state of M, except the initial state with an infinite number of tokens (figure 2-(c)). To execute the transition (q_0, a, q_1), a token is moved from q_0 to q_1 in figure 2-(d), corresponding to the configuration $(1, 0, 0)$ in instant $t = 1$. In $t = 2$, the executed transition (q_0, b, q_4) corresponds to moving a token from the initial state to a terminal one q_4 (figure 2-(e)). Since the instantaneous description does not consider neither initial states nor terminal ones, then the configuration stays the same as the previous instant. Notice that this move corresponds to both creating and terminating an instance of the FSM. Then, the transition (q_0, c, q_5) is executed by moving a token from q_0 to the hybrid state q_5. This creates a new instance implying, in this case, an increase in the number of simultaneously used instances in the execution. This is depicted in figure 2-(f). Finally, a token is moved from the state q_1 to q_2 in figure 2-(g), in order to execute the transition (q_1, a, q_2). It changes $M^{\otimes}$'s instantaneous description in $t = 4$ into $(0, 1, 1)$ which is a final configuration (i.e $(0, 1, 1) \in F_C$) since all tokens in the PCSM are in final states (either hybrid or terminal).

Formally, we define the PCSM $\mathcal{R}$ as the SM $(\Sigma_{\mathcal{R}}, \mathcal{C}_{\mathcal{R}^{\otimes}}, F_{\mathcal{C}}, c_0, \Phi_{\mathcal{R}^{\otimes}})$, where:

1. $\Sigma_{\mathcal{R}} = \bigcup_{M_j \in \mathcal{R}} \Sigma_{M_j}$;
2. $\mathcal{C}_{\mathcal{R}^{\otimes}}$ is the set of states (also called configurations of $\mathcal{R}^{\otimes}$). $\mathcal{C}_{\mathcal{R}^{\otimes}} \subset \mathbb{N}^n$, with: $n = n_I(\mathcal{R}) + n_H(\mathcal{R})$ with: $n_I(\mathcal{R})$ is the number of intermediate states in $\mathcal{R}$) ($n_I(\mathcal{R}) = |\bigcup_{M_j \in \mathcal{R}} I(M_j)|$) and $n_H(\mathcal{R})$ is the number of hybrid states in $\mathcal{R}$) ($n_H(\mathcal{R}) = |\bigcup_{M_j \in \mathcal{R}} H(M_j)|$). For each configuration c, $c[m]$ (the m^{th} component of c) is called a witness of a unique state $q_m \in Q_{M_j}$ for some j. Note that:
   - q_m is an intermediate state, if $1 \leq m \leq n_I(\mathcal{R})$;
   - q_m is an hybrid state, if $n_I(\mathcal{R}) + 1 \leq m \leq n$.

   In an abuse of notation, we use $c[m]$ and $c[q_m]$ interchangeably.
3. $F_{\mathcal{C}}$ is the set of final states. $F_{\mathcal{C}} = \{c \in \mathcal{C}_{\mathcal{R}^{\otimes}} | c[m] = 0,\ \textit{for each: } 1 \leq m \leq n_I(\mathcal{R})\}$;
4. $c_0 = \{0\}^n$ is the initial state of $\mathcal{R}^{\otimes}$;
5. $\Phi_{\mathcal{R}^{\otimes}} \subseteq \mathcal{C}_{\mathcal{R}^{\otimes}} \times \Sigma_{\mathcal{R}} \times \mathcal{C}_{\mathcal{R}^{\otimes}}$ is the set of transitions. we have $(c_1, a, c_2) \in \Phi_{\mathcal{R}^{\otimes}}$ **iff**:
   - there exists $(q_0, a, q) \in Q_{M_j}$, such that: q_0 is the initial state of M_j and $c_2[q] = c_1[q] + 1$ and $c_2[p'] = c_1[p']$ for each $p' \neq q$.
   - there exists $(p, a, q) \in Q_{M_j}$, such that: $c_2[p] = c_1[p] - 1$, $c_2[q] = c_1[q] + 1$ and $c_2[p'] = c_1[p']$ for each $p' \neq p, q$.
   - there exists $(p, a, q) \in Q_{M_j}$, such that: q is a final state or the initial state, $c_2[p] = c_1[p] - 1$ and $c_2[p'] = c_1[p']$ for each $p' \neq p$.

Simulation preorder We recall below the definition of the simulation preorder between two SMs.

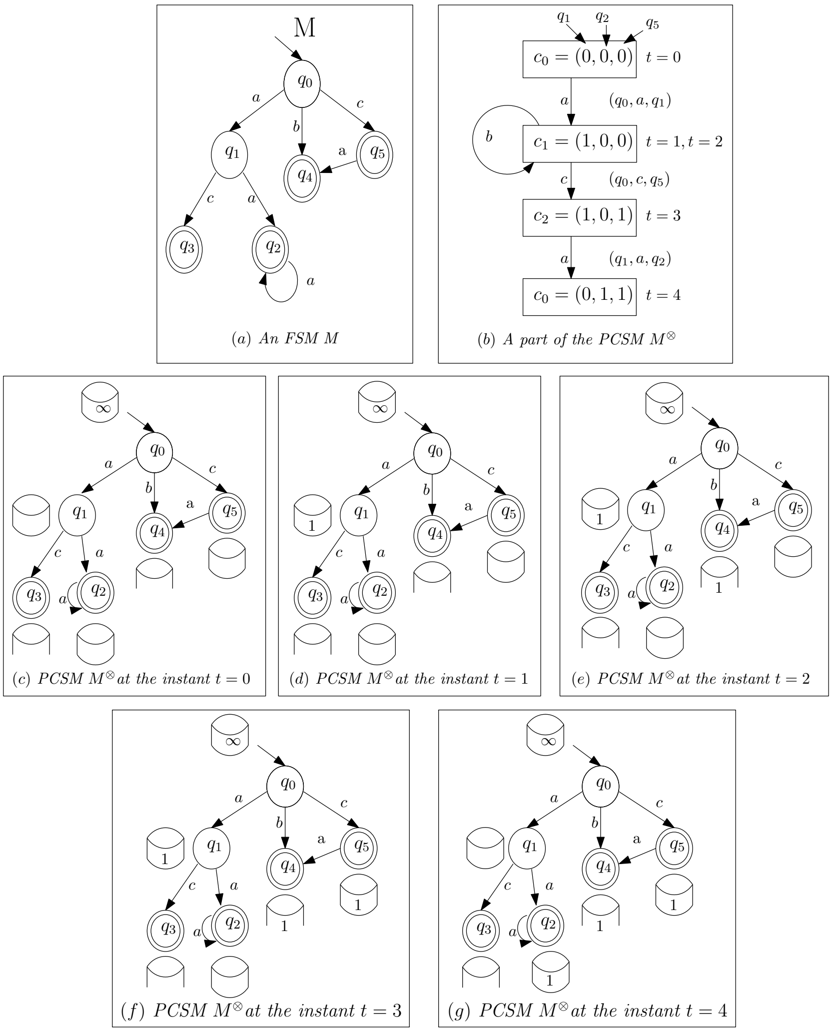

Fig. 2. An example of execution of a sequence using a PCSM.

Definition 6. ***(Simulation)***
Let $M = (\Sigma_M, Q_M, F_M, q_M^0, \delta_M)$ and $N = (\Sigma_N, Q_N, F_N, q_N^0, \delta_N)$ be two SMs. A state $p \in Q_M$ is simulated by a state $q \in Q_N$, denoted $p \ll_{(M,N)} q$ ($p \ll q$ when M and N are understood from context), iff the following two conditions hold:

1. *$\forall a \in \Sigma_M$ and $\forall p' \in Q_M$ such that $(p, a, p') \in \delta_M$, there exists $(q, a, q') \in \delta_N$ such that $p' \ll q'$, and*
2. *if $p \in F_M$, then $q \in F_N$.*

*M is simulated by N, denoted $M \ll N$, **iff** the initial state of N simulates the initial sate of M.*

Example 2. Figure 3-(c) is an example of a simulation tree, verifying if the initial state s_0 of the FSM A (figure 3-(a)) is simulated by the initial configuration $c_0 = (0, 0, 0)$ of the PCSM of M (figure 3-(b)). A branch is terminated with success when a terminal state of A is reached and paired with a final configuration (all intermidiate witnesses are null), or when a configuration of $M^{\otimes}$ that covers one of its predecessors is reached and paired with the same state of A. In this case, the simulation tree proves that $A \ll M^{\otimes}$.

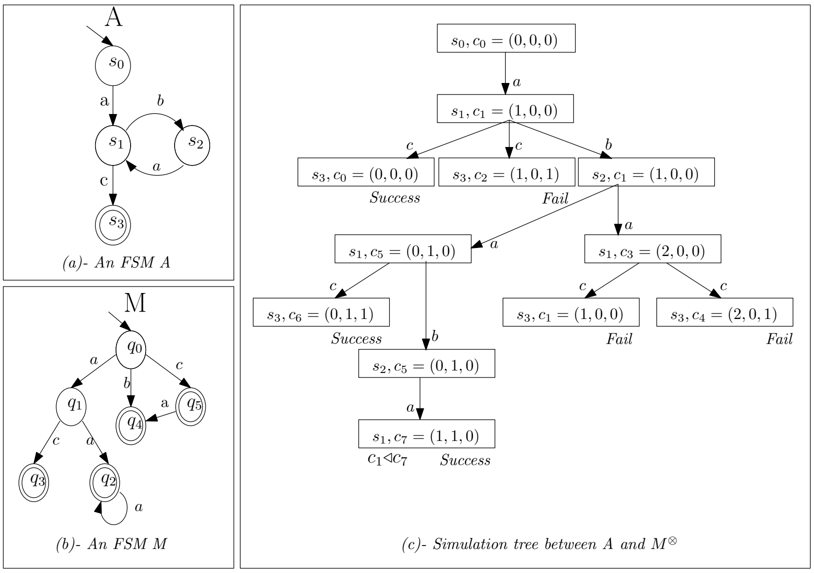

Fig. 3. An example of a simulation tree.

Interestingly, the simulation verification defined above can be seen as a two players game in a directed graph (V_a, V_d, δ, v_0), such that $V = V_a \cup V_d$ is the set of vertices with $V_a \subseteq Q_M \times Q_N$ and $V_d \subseteq Q_M \times Q_N \times \Sigma_M$, $\delta \subseteq (V_a \times V_d) \cup (V_d \times V_a)$ is the edges set verifying:
-for $(q, p) \in V_a$ and $(q, a, q') \in \delta_M$, we have $((q, p), (q', p, a)) \in \delta$; and
-for $(q, p, a) \in V_d$ and $(p, a, p') \in \delta_N$, we have $((q, p, a), (q, p')) \in \delta$.

The game is played by an attacker and a defender. It starts by putting a token in $v_0 = (q_M^0, q_N^0) \in V_a$, then the players move it along the edges of the graph. If the token is on a vertex $v \in V_a$ then the attacker moves it, otherwise it is the defender's turn.

A strategy of a player $x \in \{a, d\}$ is a function $S : V^*.V_x \mapsto V$, where $V^*.V_x$ denotes all sequences of vertices in V that end with a vertex in V_x and

$S(v_0, ..., v_k) = v_{k+1}$ implies that $(v_k, v_{k+1}) \in \delta$. In each different play, a player x adapts a strategy that decides his moves.

The defender wins every infinite play. Otherwise, the first player who can not move loses. M is simulated by N **iff** the defender has a wining strategy regardless of his opponent's strategy.

Observe that, by definition, each transition of a PCSM can at most increase or decrease a configuration component by 1. In addition, if a configuration is final then all intermediate states witnesses are equal to 0. Therefore, given a set of FSMs $\mathcal{R}$ and $c \in \mathcal{C}_{\mathcal{R}^\otimes}$, we have $\Sigma_{q \in \bigcup_{M_i \in \mathcal{R}} I(M_i)} c[q] \leq norm(c)$. Moreover, since final states can only be simulated by final ones, then for M an FSM and $p \in Q_M$, if $p \ll c$ then $norm(c) \leq norm(p)$. Hence, we are able to derive the following property.

Property 1. **(Intermediate witnesses bound) [9]** For $c \in \mathcal{C}_{\mathcal{R}^\otimes}$ and $p \in Q_M$, if $p \ll c$ then $\Sigma_{q \in \bigcup_{M_i \in \mathcal{R}} I(M_i)} c[q] \leq norm(p)$. We denote $\mathcal{C}^M_{\mathcal{R}^\otimes} = \{c \in \mathcal{C}_{\mathcal{R}^\otimes} | \Sigma_{q \in \bigcup_{M_i \in \mathcal{R}} I(M_i)} c[q] \leq norm(M)\}$.

In [9], the WSC problem in the unbounded case is reduced to simulation test between an FSM and a PCSM and this later problem is proved to be decidable. The proof of the termination of the algorithm given in [9] is based on the following property:

Property 2. **(configuration cover) [9]** Let c and c' be two configurations of $\mathcal{R}^\otimes$, such that: $c[m] = c'[m], m \in [1, n_I(\mathcal{R})]$ *and* $c[m] \leq c'[m], m \in [n_I(\mathcal{R})+1, n]$. if $q \ll c$, where q is a state of a SM M, then $q \ll c'$.

We say that c' covers c, denoted $c \lhd c'$.

We introduce below the algorithm of [9], focusing the presentation on the structure of its execution tree.

Definition 7. ***(Simulation Tree)***
We call a simulation tree $T_{sim}(M, \mathcal{R}^\otimes)$ a tree (V, v_0, E) where:

- $v_0 = (q^0_M, c_0)$ *is the root of the tree;*
- $V \subset Q_M \times \mathcal{C}^M_{\mathcal{R}^\otimes}$ *is the set of nodes;*
- *If $(q, c) \in V$ and q is final in M then so is c in $\mathcal{R}^\otimes$;*
- $E \subset V \times V$ *is the set of the tree's edges.* $\forall e = ((p, c), (q, d)) \in E : \exists a \in \Sigma_M$ *s.t* $(p, a, q) \in \delta_M$ *and* $(c, a, d) \in \Phi_{\mathcal{R}^\otimes}$*;*
- $v = (p, c) \in V$ *is a leaf in $T_{sim}(M, \mathcal{R}^\otimes)$ iff p is terminal in A or there exists an ancestor $(p, c') \in V$ of v in $T_{sim}(M, \mathcal{R}^\otimes)$ such that $c \lhd c'$.*

In the next section, we shall bound the size of this tree in the case of bounded WSC problem (i.e., when the instances of services allowed to be used in the simulation is bounded by a parameter k).

3 Bounded Composition

We call a *bounded* WSC problem, a service composition problem where the number of copies of each web service in the repository $\mathcal{R}$ used to compose the target M is bounded a priori by an integer k. This problem is formally stated as follows.

Problem 1. **Bounded Composition** $BC(M, \mathcal{R}, k)$
Input : $\mathcal{R}$ a set of FSMs; M a target FSM; k an integer.
Question : $M \ll \bigcup_{i=0}^{k} \mathcal{R}^{\otimes_i}$?

The particular case $BC(M, \mathcal{R}, 1)$ has been investigated by Muscholl and Walukiewicz [16] where it is shown to be Exptime-Complete. We shall prove in this section that $BC(M, \mathcal{R}, k)$ is also Exptime-Complete. We point out that the straightforward reduction of $BC(M, \mathcal{R}, k)$ to $BC(M, \mathcal{R}, 1)$, obtained by duplicating k times each service of $\mathcal{R}$, is not polynomial in the input size, since k may be large, and hence cannot be used to achieve our goal.

The parameter k drops the infinite aspect and reduces the search space. In this case, a loop in M can only be simulated by loops in $\mathcal{R}$. For example, one can observe that, in figure 4, S_t is not simulated by $\bigcup_{i=0}^{k}\{R_1, R_3\}^{\otimes_i}$ for every $k \in \mathbb{N}$. This is because when we repeat the loop in S_t $(k+1)$ times, there is no corresponding execution in $\bigcup_{i=0}^{k}\{R_1, R_3\}^{\otimes_i}$. However, we have $S_t \ll \bigcup_{i=0}^{k}\{R_1, R_3\}^{\otimes_i}$, for any $k \geq 1$.

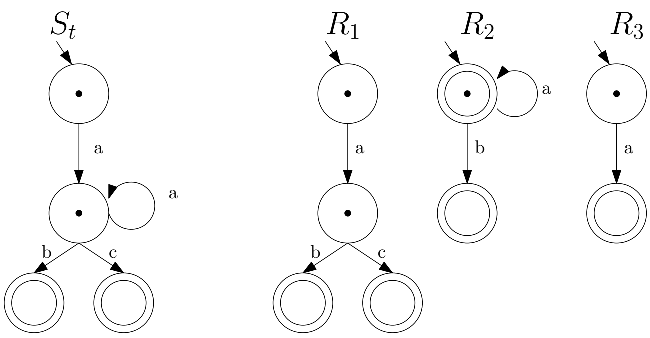

Fig. 4. A yes instance of $BC(M, \mathcal{R}, k)$ with $k = 1$.

In the following, we give an upper bound of the number of states that might appear in $\bigcup_{i=0}^{k} \mathcal{R}^{\otimes_i}$, with $k \in \mathbb{N}$.

Lemma 1. *Let $\mathcal{R}$ be a set of FSM and k is an integer. The number of states in $\bigcup_{i=0}^{k} \mathcal{R}^{\otimes_i}$ is bounded by $O(2^{nlogk})$ where $n = n_I(\mathcal{R}) + n_H(\mathcal{R})$.*

Proof. Notice that $\mathcal{R}^{\otimes} = (\bigcup_{i=0}^{k} \mathcal{R}^{\otimes_i}) \cup (\bigcup_{i=k+1}^{+\infty} \mathcal{R}^{\otimes_i})$.

In fact, the states in $\bigcup_{i=0}^{k} \mathcal{R}^{\otimes_i}$ correspond to the PCSM's configurations subset $\{c \in \mathcal{C}_{\mathcal{R}^{\otimes_k}} \mid 0 \leq c[i] \leq k, i \in [1, n]\}$. Hence, the number of states of $\bigcup_{i=0}^{k} \mathcal{R}^{\otimes_i}$ is bounded by $(k+1) \times \ldots \times (k+1) = 2^{nlog(k+1)}$. □

This lemma reduces the search space to an exponential size and leads to the following theorem.

Theorem 1. *$BC(M, \mathcal{R}, k)$ is Exptime-Complete*

Proof. `Exptime.` To show that $BC(M, \mathcal{R}, k)$ is Exptime, we bound the size of the simulation tree. A node of the simulation tree corresponds to (q, c) where q is a state of M and c a configuration of $\mathcal{R}^{\otimes_k}$. According to Lemma 1, the number of PCSM's configurations is bounded by k^n. So the number of nodes in the simulation tree is at most $|Q_M| \times k^n = 2^{nlog(k)+log(|Q_M|)}$ and therefore the complexity is in Exptime.

`Exptime-Hardness.` It can be deducted directly from the Exptime-Hardness of the particular case $BC(M, \mathcal{R}, 1)$ [16]. □

Instead of the total number of instances used in the simulation, what happens if we bound only the number of instances used simultaneously? we raise this question in the next section and prove that the problem stays Exptime-complete.

4 Bounded parallel instances

Now we consider a new parameter in service web composition that bounds the number of communications in parallel between the target and the services, i.e. the number of live services executions is bounded, but the number of instances is not. It appears that the web services composition with unbounded instances and bounded parallel instances is Exptime-Complete.

To do so, we limit the configurations of the PCSM $\mathcal{R}^{\otimes}$ to configurations where the number of waiting instances is bounded by k. Indeed, when we need to use a new instance in $\Phi_{\mathcal{R}^{\otimes}}$, we check if $\sum_{i=1}^{n} c[i] \geq k$. If so, we decrease $c[j]$ for some $j \in [n_I(\mathcal{R}) + 1, n_H(\mathcal{R})]$, i.e. we finish an instance that is waiting in an hybrid state. Let us denote by $\mathcal{R}^{\otimes_{k,p}}$ the obtained PCSM.

Problem 2. **Bounded Parallel Instances Composition** ($PBC(M, \mathcal{R}, k)$)
Input : $\mathcal{R}$ a set of FSMs;
M a target FSM.
k an integer, bounding the number of parallel instances of $\mathcal{R}$'s components used simultaneously in the simulation.
Question : $M \ll \mathcal{R}^{\otimes_{k,p}}$?

Note that $PBC(M, \mathcal{R}, k)$ can use an unbounded number of instances but only k instances in parallel.

Theorem 2. *$PBC(M, \mathcal{R}, k)$ is Exptime-complete.*

Proof. First we show that $PBC(M, \mathcal{R}, k)$ is Exptime. Clearly the entry of any configuration is bounded by k (hybrid states are included) and therefore we can check simulation in Exptime, since the depth of the simulation tree is bounded by k^n (see Lemma 1).

To show the Exptime-hardness, it suffices to note that the unbounded composition without hybrid states $UCHS(M, \mathcal{R}, 0)$ is a particular case of $PBC(M, \mathcal{R}, k)$, since we prove later in theorem 4 that $UCHS(M, \mathcal{R}, 0)$ is Exptime-hard. In fact, the number of tokens in intermediate states of $\mathcal{R}$ is bounded by $norm(M)$ (property 1). Hence, when $\mathcal{R}$ is hybrid state free, the number of instances that can be used in the simulation is bounded by $norm(M)$. In other words, it corresponds to $PBC(M, \mathcal{R}, norm(M))$. □

For k a constant, we obtain the following.

Corollary 1. *$PBC(M, \mathcal{R}, k)$ is polynomial when k is a constant.*

Proof. First of all, let us consider for every configuration c of $\mathcal{R}^{\otimes_{k,p}}$, a new component $c[n+1] = k - (\sum_{i=1}^{n} c[i])$, with $n = n_I(\mathcal{R}) + n_H(\mathcal{R})$.

For every configuration c in $\mathcal{R}^{\otimes_{k,p}}$, the non-empty witnesses $\{c[i] > 0, \leq i \leq n+1\}$ correspond to a partition of k elements (instances) into a sequence of j non empty subsets, for $j = |\{c[i] > 0, 1 \leq i \leq n\}| \leq k$. Note that j is in fact inferior to $min(k, n)$, but since k is a constant then it is more interesting to keep it as a lower bound of j.

For every $j \leq k$, the number of labeled partitions of k elements into a sequence of j non empty subsets is $j! \times \left\{{k \atop j}\right\}$, where $\left\{{k \atop j}\right\}$ is a Stirling number of the second kind [1]. Hence, the number of configurations in $\mathcal{R}^{\otimes_{k,p}}$ that have j non-empty witnesses is bounded by $C_n^j \times j! \times \left\{{k \atop j}\right\}$. Notice that $C_n^j = e^{\frac{n \ldots \times (n-j+1)}{j!}}$ is in the order of $O(n^j)$.

We conclude that the number of configurations in $\mathcal{R}^{\otimes_{k,p}}$ is bounded by $\sum_{j=1}^{k} C_n^j \times j! \times \left\{{k \atop j}\right\} \in O(n^k)$.

Finally, by applying the simulation algorithm in [10], $PBC(M, \mathcal{R}, k)$ can be decided in $O(m_v.m_e)$, where $m_v = |Q_M| + |\mathcal{C}_{\mathcal{R}^\otimes}|$ and $m_e \leq |Q_M|^2 + |\mathcal{C}_{\mathcal{R}^\otimes}|^2$ are respectively the number of edges and transitions in M and $\mathcal{R}^{\otimes_{k,p}}$. □

In the following, we show that $PBC(M, \mathcal{R}, k)$ is NP-Complete for loop-free target FSM. Let μ a sequence of letters (a word) over Σ and M the FSM that recognizes exactly μ. We call $\mu^\otimes$ the language recognized by $M^\otimes$. We consider the following NP-complete Problem [13].

Problem 3. **SHUFFLE PRODUCT**
Input : μ and μ' two words over an alphabet Σ;
Question : $\mu \in \mu'^\otimes$?

Theorem 3. *$PBC(M, \mathcal{R}, k)$ is NP-complete whenever M is loop-free.*

Proof. Clearly $PBC(M, \mathcal{R}, k)$ is in NP since the simulation relation is polynomial in the size of M. To show the NP-hardness, we reduce SHUFFLE PRODUCT to it. Let μ and μ' be an instance of SHUFFLE PRODUCT. We associate an FSM M which recognizes exactly μ and $\mathcal{R} = \{N\}$ where N is the FSM that recognizes exactly μ'. Since M is a chain, then the size of a branch of the simulation tree can not surpass $|\mu|$. Thus,the simulation verification will only explore $\mathcal{R}^{\otimes_{k,p}}$'s executions where the size is bounded by $|\mu| \leq k.|\mu'|$ with $k = \lceil \frac{|\mu|}{|\mu'|} \rceil$ and therefore the number of instances is bounded by k. Hence, $\mu \in \mu'^{\otimes}$ iff $M \ll \mathcal{R}^{\otimes_{k,p}}$ iff $M \ll \mathcal{R}^{\otimes}$. We give an example in figure 5. □

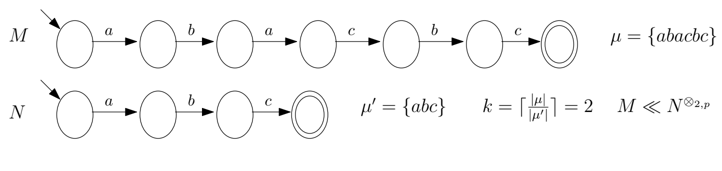

Fig. 5. An example of **SHUFFLE PRODUCT** problem.

Another factor of complexity of the WSC problem is the number of hybrid states in the available services. We investigate next the effect of this parameter on the complexity of the WSC problem.

5 Bounded number of hybrid states

The presence of hybrid states is a source of complexity in a WSC problem. As mentioned before, the size of intermediate states witnesses in configurations of $\mathcal{R}^{\otimes}$ used to simulate M is bounded by $norm(M)$. We are however unable to provide a similar bound for the number of hybrid states witnesses.

Figure 6 is an example of simulation between an FSM M and a PCSM $\mathcal{R}^{\otimes}$. The FSMs in $\mathcal{R}$ contain two hybrid states (state 1 and 2) and no intermediate state. Hence, a configuration of $\mathcal{R}^{\otimes}$ is a pair of integers witnessing the number of tokens in state 1 and state 2. The example illustrates the different roles that an hybrid state of $\mathcal{R}$ can play to simulate a state of M. Indeed an hybrid state of $\mathcal{R}$, can be used as:

(i) a terminal state, e.g., when testing whether $q_5 \ll (1,1)$, we can consider the second hybrid state of $\mathcal{R}$ as a terminal state and terminate the test, or

(ii)an intermediate state, e.g., when testing whether $q_2 \ll (1,1)$, the second hybrid state of $\mathcal{R}$ here plays the role of intermediate state, or

both a terminal and an intermediate state, e.g., when testing whether $q_1 \ll (1,0)$, a transition of $\Phi_{\mathcal{R}^{\otimes}}$ labeled by $(b,(-1,0))$ only appears in one branch in the simulation tree $\mathcal{T}_{sim}(M,\mathcal{R}^{\otimes})$. Hence, the first hybrid state of $\mathcal{R}^{\otimes}$ is considered intermediate in one branch and terminal in the other, or

a hybrid state, e.g., when it is used to simulate an hybrid state of $H(M)$.

We consider in the following the problem defined below.

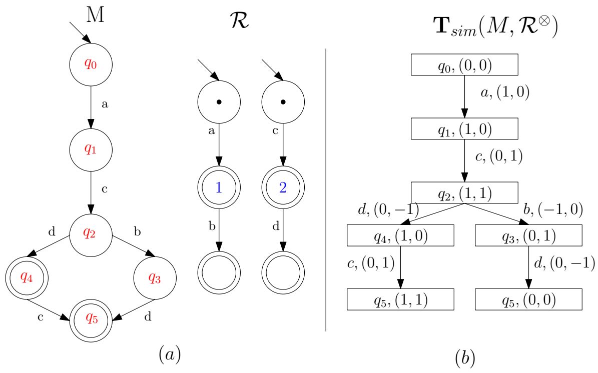

Fig. 6. Example of the simulation tree

Problem 4. **Unbounded Composition With limited number of Hybrid States** $UCHS(M, \mathcal{R}, k)$
Input : k an integer; $\mathcal{R}$ a set of FSMs, containing at most k hybrid states; M a target FSM.
Question : $M \ll \mathcal{R}^{\otimes}$?

It is worth noting that $UCHS(M, \mathcal{R}, k+1)$ is harder then $UCHS(M, \mathcal{R}, k)$. In the sequel, we progressively investigate the complexity of $UCHS(M, \mathcal{R}, k)$ problem for $k = 0$, then for $k = 1$ and finally for $k = 2$.

5.1 Case of composition without hybrid states (i.e. $k = 0$)

In this section, we are interested in the problem $UCHS(M, \mathcal{R}, 0)$. We first give a polynomial transformation, denoted $\mathcal{K}$, which is used to reduce $BC(M, \mathcal{R}, 1)$ to $UCHS(N, \mathcal{R}', 0)$. This transformation provides a mean to bound the number of instances used to prove simulation.

Definition 8. ***Transformation*** $\mathcal{K}$**.** *For an FSM* $M = (\Sigma_M, Q_M, F_M, q_0^M, \delta_M)$ *and a set of FSMs* $\mathcal{R} = \{M_1, ..., M_m\}$*, we define* $\mathcal{K}(M, \mathcal{R}) = (N, \mathcal{R}' = \{N_1, .., N_m\})$ *where:*

1. *Each* N_i *is built based on* M_i*, by adding a letter* t_i *to its alphabet, a final state* f_i *and a transition set* $\{(q_0^{M_i}, t_i, f_i)\} \cup \{(q, t_i, f_i) | q \in F_{M_i}\}$*. All final states of* M_i *become intermediate in* N_i*.*
2. *N is defined as:*
   - $\Sigma_N = \Sigma_M \cup \{t_i | 1 \leq i \leq m\}$;
   - $Q_N = Q_M \cup \{r_i | 1 \leq i \leq m\}$;
   - $F_N = \{r_m\}$;
   - $\delta_N = \delta_M \cup \{(q, t_1, r_1) | q \in F_M\} \cup \{(r_i, t_{i+1}, r_{i+1}) | 1 \leq i < m\}$.

Figure 7 illustrates an example of this transformation. We prove later in proposition 2 that $\mathcal{K}$ defines a polynomial reduction of $BC(M, \mathcal{R}, 1)$ to $UCHS(N, \mathcal{R}', 0)$. In fact, the intuition behind this reduction is based on two points:

- By adding the sequence of letters $t_1, ..., t_m$ at the end of every execution accepted by N and adding t_i at the end of every execution accepted by $N_i \in \mathcal{R}'$, we ensure that even in an unbounded instances simulation, we can not use more than one instance of every N_i in order to simulate N.
- The construction of $\mathcal{R}'$ verifies that every hybrid state in $M_i \in \mathcal{R}$ becomes intermediate in N_i, while keeping its dual role: either terminate the execution by adding the letter t_i to the execution of $N_i$and reaching the terminal state f_i, or keep the execution in the same way as M_i.

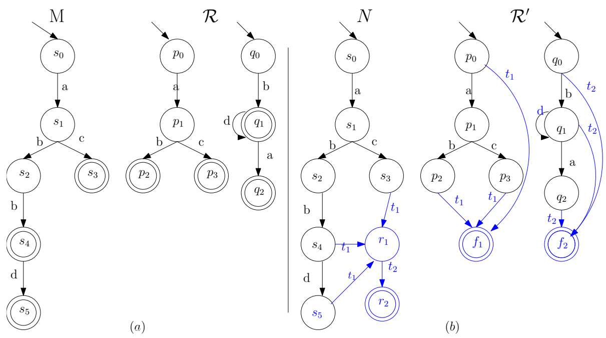

Fig. 7. *An example of transformation* $\mathcal{K}$

The following propositions show that the transformation $\mathcal{K}$ preserves the simulation preorder.

Proposition 1. *Let M be an FSM, $\mathcal{R} = \{M_1, ..., M_m\}$ be a set of FSMs and $\mathcal{K}(M, \mathcal{R}) = (N, \mathcal{R}' = \{N_1, .., N_m\})$. For p and q two states of respectively M and $\mathcal{R}^{\otimes_1}$, we have: $p \ll_{(M,(\mathcal{R})^{\otimes_1})} q$ iff $p \ll_{(N,(\mathcal{R}')^{\otimes_1})} q$.*

Proof. By construction of $\mathcal{K}(M, \mathcal{R})$, *if* $p \ll_{(M,\mathcal{R}^{\otimes_1})} q$ and p is terminal in M *then* $p \ll_{(N,(\mathcal{R}')^{\otimes_1})} q$.

We suppose next that:

If $(p, a, p') \in \delta_M$, $(q, a, q') \in \delta_{\mathcal{R}^{\otimes_1}}$ and $p' \ll_{(M,\mathcal{R}^{\otimes_1})} q'$, *then* $p' \ll_{(N,(\mathcal{R}')^{\otimes_1})} q'$.

and prove that $p \ll_{(N,(\mathcal{R}')^{\otimes_1})} q$.

For each $(p, a, p') \in \delta_N$, we have:

- *if* $a \in \Sigma_M$, *then* there exists $(q,a,q') \in \delta_{\mathcal{R}^{\otimes_1}} \subseteq \delta_{(\mathcal{R}')^{\otimes_1}}$ such that $p' \ll_{(N,(\mathcal{R}')^{\otimes_1})} q'$.
- else $a = t_1$, $p' = r_1$ and q is a product of final states of $\mathcal{R}$. therefore, there exists $(q,t_1,q') \in \delta_{(\mathcal{R}')^{\otimes_1}}$ such that $q' = (f_1, {q'}_{i_1}, ..., {q'}_{i_l})$ where ${q'}_{i_j}$ is final in $\mathcal{R}$ such that $p' \ll_{(N,(\mathcal{R}')^{\otimes_1})} q'$.

We conclude that if $p \ll_{(M,\mathcal{R}^{\otimes_1})} q$ then $p \ll_{(N,(\mathcal{R}')^{\otimes_1})} q$.

Reciprocally, we have $(p,a,p') \in \delta_N$ (respectively $\delta_{(\mathcal{R}')^{\otimes_1}}$) and $a \notin \{t_i | 1 \leq i \leq m\}$ iff $(p,a,p') \in \delta_M$ (respectively $\delta_{\mathcal{R}^{\otimes_1}}$). In addition, the definition of $\mathcal{K}$ ensures that if p is final in M and $p \ll_{(N,(\mathcal{R}')^{\otimes_1})} q$ then q is final in $\mathcal{R}^{\otimes_1}$. Hence if $p \ll_{(N,(\mathcal{R}')^{\otimes_1})} q$ then $p \ll_{(M,\mathcal{R}^{\otimes_1})} q$. □

In particular, we take p as the initial state of M and q the initial state of $\mathcal{R}^{\otimes_1}$. This implies that:

Proposition 2. *Let M be an FSM, $\mathcal{R} = \{M_1, ..., M_m\}$ be a set of FSMs and $\mathcal{K}(M,\mathcal{R}) = (N, \mathcal{R}' = \{N_1, .., N_m\})$. We have: $M \ll \mathcal{R}^{\otimes_1}$ iff $N \ll (\mathcal{R}')^{\otimes}$.*

Proof. We have $N \ll (\mathcal{R}')^{\otimes_1}$ *iff* $N \ll (\mathcal{R}')^{\otimes}$. Indeed, each path that starts from the initial state to a final one in N contains exactly one transition labelled by t_i, for each $i \in [1,m]$ and a similar path in each N_i contains exactly one transition labelled by t_i. □

Hence, $\mathcal{K}$ is a polynomial reduction of $BC(M,\mathcal{R},1)$ problem to the UCHS problem. This enables to derive the following result.

Theorem 4. *$UCHS(M,\mathcal{R},0)$ problem is Exptime-complete.*

Proof. According to proposition 2, the $\mathcal{K}$ transformation reduces $BC(M,\mathcal{R},1)$ to $UCHS(M,\mathcal{R},0)$ in polynomial time. Thus $UCHS(M,\mathcal{R},0)$ is Exptime-hard. Since it is also proven Exptime in [9], then $UCHS(M,\mathcal{R},0)$ is Exptime-complete. □

5.2 Case of composition with one hybrid state

We consider the problem $UCHS(M,\mathcal{R},1)$ where M is an FSM and $\mathcal{R}$ a set of FSMs containing at most one hybrid state ($n_H(\mathcal{R}) \leq 1$). We denote $k_0 = |Q_M|.2^{n_I(\mathcal{R}).log(norm(M))}$. Two nodes (q,c) and (q',c') in a simulation tree are called comparable if $q = q'$ and either $c \lhd c'$ or $c' \lhd c$. The nodes (q,c) and (q',c') are said incomparable otherwise.

Property 3. Let $\mathcal{R}$ be a set of FSMs containing at most one hybrid state. Two configurations of $\mathcal{R}^{\otimes}$ are comparable by the cover relation, *iff* they have exactly the same intermediate witnesses.

Proof. Acoording to property 2, for c, c' two configurations in $\mathcal{R}^{\otimes}$ we have $c \lhd c'$ *iff*:

1. c and c' have the same intermediate witnesses; and

2. for every hybrid witness $c[h]$, we have: $c[h] \leq c'[h]$.

In the current case, we consider that $\mathcal{R}$ has at most one hybrid witness. Hence, for any pair of configurations of $\mathcal{R}^{\otimes}$, condition 2 is verified.

We conclude that for every two configurations c, c' in $\mathcal{R}^{\otimes}$, $c \lhd c'$ *iff* c and c' have the same intermediate witnesses. □

Property 4. Let S be a set of nodes of $\mathcal{T}_{sim}(M, \mathcal{R}^{\otimes})$ that are pairwise incomparable, *then* $|S| \leq k_0$.

Proof. In configurations considered in $\mathcal{T}_{sim}(M, \mathcal{R}^{\otimes})$, intermediate witnesses are bounded by $norm(M)$ (property 1). Therefore and according to property 3, the number of incomparable configurations considered in $\mathcal{T}_{sim}(M, \mathcal{R}^{\otimes})$ is at most $2^{n_I(\mathcal{R}).log(norm(M))}$. Since $S \subset Q_M \times \mathcal{C}_{\mathcal{R}^{\otimes}}$, then $|S| \leq k_0$. □

Proposition 3. *If* $n_H(\mathcal{R}) \leq 1$, *then foreach* $(q, c) \in \mathcal{T}_{sim}(M, \mathcal{R}^{\otimes})$, $c[h = (n_I(\mathcal{R}) + 1)] \leq {k_0}^2$.

Proof. let P be a path in $\mathcal{T}_{sim}(M, \mathcal{R}^{\otimes})$, Int be an interval in $\mathbb{N}$ and $S = (v_n = (q_n, c_n))_{n \in Int}$ be a sequence of nodes in P such that:
-v_i is the i^{th} node met in P that is comparable to one of its predecessors $v = (q_i, c)$; and
-For each $i, j \in Int$, v_i and v_j are incomparable.

If $Int = \emptyset$, then all nodes of P are not comparable. The size of P is then bounded by k_0, therefore, $c[n_I(\mathcal{R}) + 1] \leq k_0$ for each (q,c) in P.

We suppose next that $Int \neq \emptyset$ and take $Int = [1, k]$, $k \in \mathbb{N}$. We prove recursively that for each $l \in [1, k]$, $c_l[h] \leq l.k_0$.

For $l = 1$, we have $c_1[n_I(h] \leq k_0$.
For $1 < l < k$, we suppose that $c_l[h] \leq l.k_0$. Each node $v = (q, c)$ between v_l and v_{l+1} in P is either:

1. comparable to a node v_i with $i \in [1, l]$. In this case, $c[h] < c_i[h] \leq l.k_0$ (otherwise v should be a leaf).
2. incomparable to all its predecessors. The number of such nodes is bounded by k_0. And since transitions displacements is in $\{-1, 0, 1\}^h$, then we have $c[h] < l.k_0 + k_0$.

Therefore $c_{l+1}[h] \leq (l + 1).K_0$.

Once we reach v_k, each one of its possible successors $v = (q, c)$ is comparable to a node v_i with $c[h] < c_i[h]$, except for the last one that is the leaf of P.

Finally, since $k < k_0$ (because S is a sequence of incomparable nodes), we conclude that each node of P is in $Q_A \times ([1, norm(A)]^I \times [1, {k_0}^2])$. □

Since deciding simulation only requires to visit a node once, we argue next that this problem is in APspace (i.e a problem that can be solved by an alternating Turing machine in polynomial space): the size of a position of the simulation tree is polynomial in the input size (Proposition 3). Hence a polynomial space alternating turing machine can solve this simulation problem: universal states correspond to the target's and existential states correspond to the shuffle product's configurations. Note that APspace corresponds to Exponential time complexity. Given the above, we conclude that:

Lemma 2. *$UCHS(M, \mathcal{R}, 1)$ is in Exptime.*

To prove the Exptime-hardness of the problem, we recall that $UCHS(M, \mathcal{R}, 0)$ is Exptime-hard (theorem 4) and that $UCHS(M, \mathcal{R}, 1)$ is harder than $UCHS(M,$ $\mathcal{R}$, 0).

Theorem 5. *$UCHS(M, \mathcal{R}, 1)$ is Exptime-complete.*

5.3 Case of composition with two hybrid states

In this section, we consider the problem of unbounded composition of web services with at most 2 hybrid states in $\mathcal{R}$, *i.e.* $UCHS(M, \mathcal{R}, 2)$. Our approach is based on relating this simulation problem to the reachability issue.

For $x \in \mathbb{N}^{i_1}$ and $y \in \mathbb{N}^{i_2}$, we denote the concatenation of two vectors x and y, $(x.y) \in \mathbb{N}^{i_1+i_2}$ such that:

$$(x.y)[j] = \begin{cases} x[j] \ if \ j \in [1, i_1] \\ y[j] \ if \ j \in [i_1+1, i_1+i_2] \end{cases}$$

We define the 2-dimension Vector Addition System with States (VASS) [11] $\mathcal{V}_{M,\mathcal{R}}$ as follows:

- States: $S \subseteq Q_M \times \{c \in \mathbb{N}^{n_I(\mathcal{R})} | \sum_{i=1}^{i=n_I(\mathcal{R})} c[i] \leq norm(M)\}$.
- Transitions: $W \subseteq S \times S \times \{-1, 0, 1\}^2$ such that: $((q,c),(p,d),x) \in W$ *iff* there exists $a \in \Sigma_M$, $y \in \mathbb{N}^2$ such that $(p,a,q) \in Q_M$ *and* $((c,y),a,(d,y+x)) \in \Phi_{\mathcal{R}^\otimes}$ and $\sum_{i=1}^{i=n_I(\mathcal{R})} c[i] \leq norm(M)$ and $\sum_{i=1}^{i=n_I(\mathcal{R})} d[i] \leq norm(M)$.
- Initial configuration: the system starts with the state $(q_0^M, \{0\}^{n_I(\mathcal{R})})$ and the vector $(0,0)$.

Figure 8 depicts an example of a VASS associated to an FSM M and a set of FSMs $\mathcal{R}$.

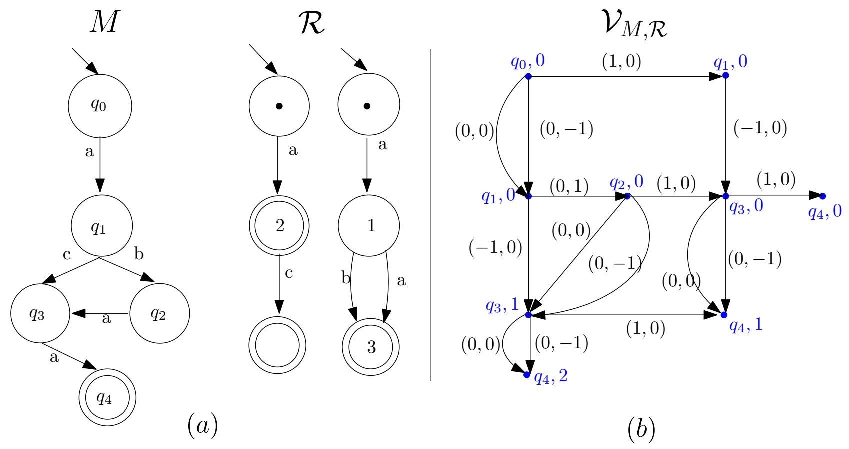

Fig. 8. *An example of a VASS associated to an FSM M and an FSMs set.*

The reachability issue in 2-dimension VASSs has been investigated by Hopcroft and Pansiot [11] in the general case where displacements are in $\mathbb{N}^2$. [11] gives an algorithm to prove the semi-linearity of the reachability set of such systems. The algorithm builds a tree T_{reach} labeled by 3-tuples (p, c, A_c) where p is the current state, $c \in \mathbb{N}^2$ is a vector reached in the system and $A_c \subset \mathbb{N}^2$. (p, c, A_c) denotes that every vector in the linear set $\{c + \alpha_1 a_1 + ... + \alpha_n a_n | i \in [1, n], a_i \in A_c \text{ and } \alpha_i \in \mathbb{N}\}$ can be reached in state p from the initial configuration.

We consider in the following a simulation tree $T_{sim}(M, \mathcal{R}^{\otimes})=(V, v_0, E)$, a recheability tree $T_{reach}(\mathcal{V}_{M,\mathcal{R}})=(V', v'_0, E')$ and a function π defined as follows:

$$\begin{array}{rccc} \pi : & V' & \rightarrow & V \\ & ((p, c), x, A_x) & \mapsto & (p, (c.x)) \end{array}$$

The following proposition enables to establish a connection between paths in a simulation tree and a corresponding recheability tree.

Proposition 4. *Let $\mu = v_0...v_t$ be a path in $T_{sim}(M, \mathcal{R}^{\otimes})$. Then there exists a path $\mu' = v'_0...v'_t$ in $T_{reach}(\mathcal{V}_{M,\mathcal{R}})$ such that $v_i = \pi(v'_i)$, $i \in [0, t]$.*

Proof. We proof by induction on the length i of the path $\mu = v_0...v_t$.

For $i = 0$ we have $v_0 = (q_0^M, c_0) = \pi((q_0^M, \{0\}^{n_I(\mathcal{R})}), x_0, A_{c_0}) = \pi(v'_0)$.

Now suppose that the property is true for $i < t$ and $v_0...v_{i+1}$ is a path in $T_{sim}(M, \mathcal{R}^{\otimes})$. Then by hypothesis there exists a path $v'_0...v'_i$ in $T_{reach}(\mathcal{V}_{M,\mathcal{R}})$, such that $v_j = \pi(v'_j)$, $j \in [0, i]$.

Suppose that v'_i is a leaf in $T_{reach}(\mathcal{V}_{M,\mathcal{R}})$. Then according to the algorithm of Hopcroft and Pansiot [11], we have either:

- There exist $j \in [0, i-1]$ such that $v'_j = ((p,c), y, A_x)$ and $v'_i = ((p,c), x, A_x)$ with $y \leq x$ (see Algorithm **??**, line 1). This implies that $v_j \lhd v_i$, which contradicts that v_i is not a leaf in $T_{sim}(M, \mathcal{R}^{\otimes})$, i.e. $(c,y) \lhd (c,x)$.
- There is no transition from v'_i in the system (see Algorithm **??**, line 1). But for $v_i = (p,(c,x))$ and $v_{i+1} = (q,(d,y))$ we have $v_i v_{i+1} \in E$ which means that $(p,a,q) \in \delta_M$ and $((c,x), a, (d,y)) \in \Phi_{\mathcal{R}^{\otimes}}$. This implies that $((p,c),(q,d), y-x) \in W$. Contradiction.

Therefore, we have : $v'_{i+1} = ((q,d), y, A_y)$ is a successor of v'_i in $T_{reach}(\mathcal{V}_{M,\mathcal{R}})$, with $v_{i+1} = (q,(d,y))$. We conclude that μ' is a path in $T_{reach}(\mathcal{V}_{M,\mathcal{R}})$. □

The following corollary is a consequence of Proposition 4.

Corollary 2. *$T_{sim}(M, \mathcal{R}^{\otimes})$ is a sub-tree of $T_{reach}(\mathcal{V}_{M,\mathcal{R}})$.*

Clearly the time complexity for computing $T_{sim}(M, \mathcal{R}^{\otimes})$ is dominated by the complexity of computing $T_{reach}(\mathcal{V}_{M,\mathcal{R}})$. Moreover we know from [12] that the size of $T_{reach}(\mathcal{V}_{M,\mathcal{R}})$ is in 2-Exptime. Hence, we derive the following complexity result.

Theorem 6. *$UCHS(M, \mathcal{R}, 2)$ is in 3-Exptime.*

Proof. According to [12], the size of $T_{reach}(\mathcal{V}_{M,\mathcal{R}})$ is of order $O(2^{2^{\alpha}})$ where $\alpha = max(|S|, |W|) \leq c \times (|Q_M| \times Norm(M)^{n_I(\mathcal{R})})^2$ with c is a constant. Then according to Corollary 2, the size of $T_{sim}(M, \mathcal{R}^{\otimes})$ is bounded by $2^{2^{2^{c_1+c_2*\beta}}}$ where c_1 and c_2 are constants and $\beta = log(|Q_M|) + n_I(\mathcal{R}) \times log(Norm(M))$. □

Our proof for Theorem 6 can be seen more as an embedding of the search space explored by a simulation test to the one explored when the reachability issue is considered. This is an approach that can not so far be generalized because the best upper bound provided for vector addition systems reachability is non-primitive recursive; in fact even the existence of a primitive upper-bound is still open [14].

6 Conclusion

In this paper we have considered two parameters that are source of complexity of the web services composition problem. We have shown that among the considered problems, several instances remain Exptime-complete when a parameter is bounded. It remains an open question to identify the complexity of $UCHS(M, \mathcal{R}, k)$ for any $k \in \mathbb{N}$; [5] proves in the context of Z-Reachability that the problem is k-Exptime. This complexity is quite far from the known lower bound (2-Exptime). It is also interesting to improve the polynomial complexity given for k=2 in [7] (polynomial of the 17^{th} degree) and/or give a simpler algorithm that can eventually be extended to the general case.

References

1. Milton Abramowitz and Irene A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables.* Dover, New York, ninth dover printing, tenth gpo printing edition, 1964.
2. Gustavo Alonso, Fabio Casati, Harumi Kuno, and Vijay Machiraju. *Web Services: Concepts, Architectures and Applications.* Springer, 2010.
3. B. Benatallah, F. Casati, and F. Toumani. Web Service Conversation Modeling: A Cornerstone for E-Business Automation. *IEEE Internet Computing*, 08(1):46–54, 2004.
4. D. Berardi, D.Calvanese, G. De Giacomo, R. Hull, and M. Mecella. Automatic composition of transition-based semantic web services with messaging. In *VLDB*, pages 613–624, 2005.
5. Tomáš Brázdil, Petr Jančar, and Antonín Kučera. Reachability games on extended vector addition systems with states. In *Automata, Languages and Programming*, pages 478–489. Springer, 2010.
6. T. Bultan, X. Fu, R. Hull, and J. Su. Conversation specification: a new approach to design and analysis of e-service composition. In *WWW'03*. ACM, 2003.
7. Jakub Chaloupka. Z-reachability problem for games on 2-dimensional vector addition systems with states is in p. In *Reachability Problems*, pages 104–119. Springer, 2010.
8. Jean-Baptiste Courtois and Sylvain Schmitz. Alternating vector addition systems with states. In *Mathematical Foundations of Computer Science 2014*, pages 220–231. Springer, 2014.
9. Ramy Ragab Hassen, Lhouari Nourine, and Farouk Toumani. Protocol-based web service composition. In *ICSOC*, pages 38–53, 2008.
10. Monika Rauch Henzinger, Thomas A Henzinger, and Peter W Kopke. Computing simulations on finite and infinite graphs. In *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on*, pages 453–462. IEEE, 1995.
11. John Hopcroft and Jean-Jacques Pansiot. On the reachability problem for 5-dimensional vector addition systems. *TCS*, 8(2):135–159, 1979.
12. Rodney R Howell, Louis E Rosier, Dung T Huynh, and Hsu-Chun Yen. Some complexity bounds for problems concerning finite and 2-dimensional vector addition systems with states. *TCS*, 46:107–140, 1986.
13. Joana Jedrzejowicz and Andrzej Szepietowski. Shuffle languages are in p. *Theoretical Computer Science*, 250(1-2):31–53, 2001.
14. S Rao Kosaraju. Decidability of reachability in vector addition systems. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, pages 267–281. ACM, 1982.
15. Slawomir Lasota. Expspace lower bounds for the simulation preorder between a communication-free petri net and a finite-state system. *Inf. Process. Lett.*, 109(15):850–855, 2009.
16. Anca Muscholl and Igor Walukiewicz. A lower bound on web services composition. *Logical Methods in Computer Science*, 4(2), 2008.