23

Finitely Representing Infinite Reachability Graphs of CFSMs with Graph Grammars

Quemener, Y.-M. and Jéron, T.
IRISA
Campus de Beaulieu, 35042 Rennes CEDEX, France.
E-mail: {quemener, jeron}@irisa.fr

Abstract

We propose here an algorithm enabling to represent, in a finite way, some infinite reachability graphs of communicating finite-state machines, by using a graph grammar. The model-checking algorithm presented in Burkart and Quemener (1996) uses that finite representation for verifying properties of the infinite graph. In way to obtain that finite representation, we use a result of Jéron and Jard (1993): it can be detected that some sequences of transitions are infinitely repeated. We show here that the transitions issued from states linked by such sequences are also infinitely repeated if they are repeated twice. We deduce a method for detecting patterns that compose the infinite reachability graph on study.

Keywords

Infinite-state systems, communicating finite-state machines, graph grammars, verification.

1 INTRODUCTION

Verification of distributed systems is crucial for ensuring the safety of decentralized informatics. Works on that topic focused, until recently, on the study of finite-state systems. This can be easily understood, as the finiteness of the reachability graph implies decidability results about the model-checking of temporal logics. Quite recently, such results have been proved for some infinite-state systems (cf. Muller and Schupp (1985) and Courcelle (1990) for example). Since then, some algorithms have been proposed for deciding model-checking (cf. Burkart and Steffen (1992), Purushothaman Iyer (1993) and Hungar and Steffen (1993) for example) for various classes of infinite-state systems. Other works have focussed on bisimulation equivalence for such infinite-state systems (cf. Baeten, Bergstra and Klop (1987) and Hüttel and Stirling (1991) for example).

The works proposed so far study infinite-state systems defined in terms of context-free or pushdown processes, i.e. terms of a process algebra which are such that reachability graphs

R. Gotzhein et al. (eds.), *Formal Description Techniques IX*

have decidable properties. We are interested in the verification of distributed systems described as **communicating finite-state machines,** a model introduced by Bochmann (1978), i.e. finite automata communicating by sending messages through FIFO queues (in short, CFSMs). That model underlies the formal description techniques Estelle (cf. ISO (1989)) and SDL (cf. CCITT (1988)), used for the specification of communication protocols. Unfortunately, that model has the same expressive power as Turing machines, hence it is for example undecidable to know whether the reachability graph of a system of two CFSMs is infinite when the communication channels are unbounded as has been shown by Brand and Zafiropoulo (1983).

The verification methodology for CFSMs can then be roughly described as follows. First, try to build the reachability graph. If you obtain a result, you can use verification tools on the finite reachability graph. If the construction of the reachability graph does not succeed, it means that the reachability graph is either too big, or infinite. You can then use simulation methods, or artificially limit the size of the communication queues for obtaining a finite reachability graph. A first improvement on that situation is given by Jéron and Jard (1993) who propose a semi-decision method of the unboundedness problem by detecting, during the construction of the reachability graph, transition sequences that can be infinitely repeated and that increase a communication queue, hence resulting in an unbounded queue and an infinite reachability graph.

We want to continue that work for providing formal and exhaustive verification methods for CFSMs with an infinite reachability graph. A first step is to obtain a representation of the reachability graph which enables to use methods adapted to infinite graphs. This is what we study in this paper. We extend the results of Jéron and Jard (1993) in the following way: we show that the transitions issued from repeated states are the same for all those states if this is true for the first two states. In that way, the sequences known to be repeatable form the skeleton of a pattern, and the transitions between those sequences, whose it can be tested whether they are repeatable, complete a pattern which is infinitely repeated for building the reachability graph. In other words, we propose to test whether the infinite reachability graph of CFSMs can be finitely represented as a combination of identical, finite patterns, repeating infinitely, i.e. if that infinite graph is generated by a given graph grammar (see for example Caucal (1992) for a presentation of graph grammars). Quemener and Jéron (1995) and Burkart and Quemener (1996) propose two algorithms that perform the model-checking of such infinite graphs for CTL, by using their finite representation in the form of a graph grammar.

The rest of the paper is organized as follows. We first give some definitions about CFSMs, and the main results of Jéron and Jard (1993). We then introduce graph grammars, and present new results enabling to distinguish patterns in an infinite reachability graph. Finally, we give a sketch of our proposed algorithm. Some proofs are omitted, and can be found in Quemener and Jéron (1996).

2 CFSMS AND UNBOUNDEDNESS TEST

2.1 Notations and definitions

We begin by some basic notions on words.

Let A be a finite **alphabet**, the elements of which are called **letters**. A finite sequence $x = (a_1, \ldots, a_n)$ of letters is called a **word** and is denoted $a_1 \ldots a_n$. A^* is the set of all the words over A, and ϵ is the empty word. We note $|x|$ the length of a word x, and we have $|\epsilon| = 0$. The set of p-uples of words over the alphabets $A_1, \ldots, A_p$ is the cartesian product $\times_{i=1}^{i=p} A_i^*$.

The **concatenation** of two words is denoted in the usual way by a dot. The concatenation is generalized to p-uples of words by component-wise concatenation.

If $z = x.y$ we say that x (resp. y) is a **left** (resp. **right**) **factor** of z. The left factor usually defines a partial ordering $\leq$ on words, called the **prefix ordering**, generalized to p-uples of words:

Definition 1 *Let $x, y \in A^*$ and $X =< x_1, \ldots, x_p >$, $Y =< y_1, \ldots, y_p >$ be two p-uples of words. Then:*

$x \leq y$ *if* *x is a left factor of y.* $\quad X \leq Y$ *if* $\forall i \in [1..p], x_i \leq y_i$.

If n is a positive integer, x^n is the concatenation of n words equal to x.

We now present a specialized form of a transition system, introduced by Jéron and Jard (1993), that can model the behaviour of a distributed system using FIFO channels, in particular a Cfsm system (cf. Bochmann (1978)).

A **transition system** is a 4-uple $\mathcal{S} =< GS, T, \rightarrow, S_0 >$. GS is the set of **global states**, T is a finite set of **transitions**, the **transition function** $\rightarrow$ is a partial function from $GS \times T$ to GS and $S_0 \in GS$ is the initial state.

A global state $S \in GS$ is an $(N + M)$-uple $S =< E(S), C(S) >$ where :

- $E(S) =< E_1(S), \ldots, E_N(S) >$ is the N–uple of **local states**. For all i in $[1..N]$, E_i is a function from GS to a finite set LS_i.
- $C(S) =< C_1(S), \ldots, C_M(S) >$ is the M–uple of **channel contents**. For all j in $[1..M]$, C_j is a function from GS to M_j^*. The elements of the finite alphabet M_j are called **messages**.

A transition $t \in T$ is a triple $t =< \delta(t), in(t), out(t) >$ where :

- $\delta(t) =< \delta_1(t), \ldots, \delta_N(t) >$ and each **local transition function** $\delta_i(t)$ is a partial function from LS_i to LS_i.
- $in(t) =< in_1(t), \ldots, in_M(t) >$ and $out(t) =< out_1(t), \ldots, out_M(t) >$ respectively represent the **channels inputs** and **outputs**. For all j in $[1..M]$, in_j and out_j are functions from T to M_j^*.

The transition $t \in T$ is **fireable** in a global state S iff :

- $\delta(t)(E(S))$ is defined, i.e. all local transitions are fireable : $\forall i \in [1..N], \delta_i(t)(E_i(S))$ is defined,
- $in(t) \leq C(S)$ i.e. all receptions are posssible : $\forall j \in [1..M], in_j(t) \leq C_j(S)$.

This transition then leads to a state S'. We write $S \xrightarrow{t} S'$ and we have :

- $E(S') = \delta(t)(E(S))$ (change of local states),
- $in(t).C(S') = C(S).out(t)$ (change of channel contents in a FIFO way).

The transition function $\rightarrow$ is extended to transition sequences of T^* by :
$S_1 \xrightarrow{t_1 \ldots t_n}_* S_{n+1}$ iff there exist $n-1$ states $S_2, \ldots, S_n$ s.t. $\forall i \in [1..n]$, $S_i \xrightarrow{t_i} S_{i+1}$
The two functions *in* and *out* are extended into functions on sequences of T^*. Each coordinate in_j (resp. out_j) is extended into a morphism of monoids from T^* to M_j^*.

A global state $S \in GS$ is called **reachable** (from the initial state S_0) if there exists a transition sequence $w \in T^*$ such that $S_0 \xrightarrow{w}_* S$. For simplicity reasons, we will interest ourselves in transition systems such that all states $S \in GS$ are reachable. In that way, a transition system is isomorphic to its reachability graph, and, in the following, we will use the two words without distinction.

The above partial ordering on global states generalizes the prefix ordering on words :

Definition 2 *S is a generalized prefix of S' denoted $S \leq_g S'$ if the local states of S and S' are identical, and channels contents in S are prefixes (on words) of those of S' i.e.* $S \leq_g S' \iff (E(S) = E(S')$ *and* $C(S) \leq C(S'))$

We will note $S <_g S'$ if $S \leq_g S'$ and $S \neq S'$ (thus $\exists f_j$ such that $C_j(S) < C_j(S')$).

A system of Cfsm is an N-uple $< P_1, \ldots, P_N >$ of finite state machines which communicate asynchronously by messages. Each Cfsm P_i is a 4-uple $< LS_i, T_i, \delta_i, q_{0_i} >$ where LS_i is the finite set of local states, T_i is a finite set of transitions (a transition t is an output $-a, a \in M_{ij}$ or an input $+b, b \in M_{ji}$ or an internal transition e), δ_i is a partial function from $LS_i \times T_i$ to LS_i, and $q_{0_i} \in LS_i$ is the initial state. For each pair $(P_i, P_j), i \neq j$ of Cfsm, there exists a Fifo channel f_{ij} which allows the communication of messages in M_{ij} from P_i to P_j.

The behaviour of such a system is described by a restrictive form of our transition system. A global state is an $N + (N^2 - N)$–uple of local states and channel contents. The initial state consists in all the local initial states and empty channels. A transition of the system is an element $t \in T = \bigcup_{i=1}^{N} T_i$. A transition is either an input or an output or an internal transition.

We now give an example. First, the Figure 0.1 depicts a CFSM, modeling a (flawed) connect-disconnect protocol. The left automaton can demand the opening of a session, by emitting the message a (transition labelled $-a$). After reception of that message by the right automaton (transition labelled $+a$), either of the two automata can demand the closure of the session (transitions labelled $-b$ and $-c$). All messages are received in the FIFO queues f_1 and f_2, attached to each automaton.

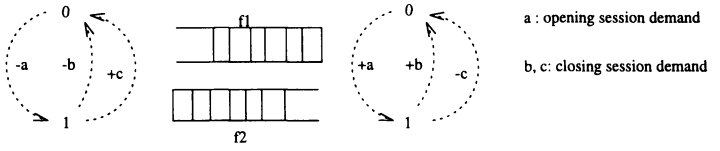

Figure 1 An example of a CFSM: connect-disconnect protocol.

The reachability graph of that CFSM is given in Figure 0.2. A global state of the system

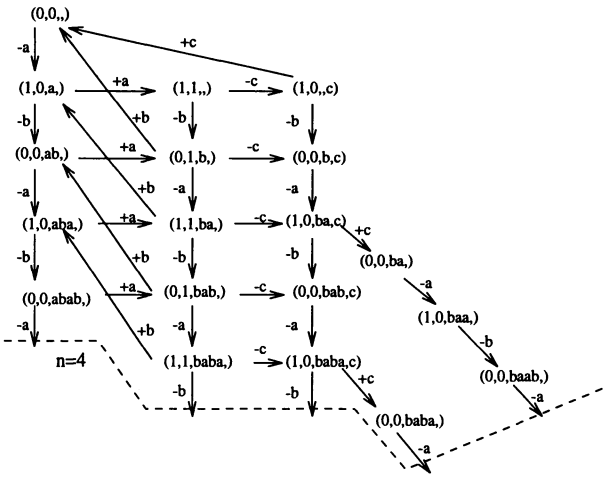

Figure 2 Reachability graph of the CFSM depicted in Figure 0.1.

is a quadruple: the two first components are the states of the automata, and the two last are the contents of the FIFO queues. We have limited the graph to the part where FIFO queues have less than four elements, but, if we assume unbounded queues, the graph is infinite and can be represented by a graph grammar, as we will see in section 3.

2.2 The unboundedness test

Jéron and Jard (1993) propose a binary relation $\mathcal{U}_{\neq}$ between states of a transition system such that the sequence of transitions linking the two states can be infinitely repeated, hence implying that there is an infinite number of global states in $\mathcal{S}$.

Definition 3 *If S and S' are two global states of $\mathcal{S}$, such that $S \xrightarrow{w}_* S'$, we note:*

$\mathcal{U}(S, S')$ *if* $E(S) = E(S')$ *and* $C(S).out(w) \leq C(S').out(w)$

We will note $\mathcal{U}_{\neq}(S, S')$ if we have $\mathcal{U}(S, S')$ and $S \neq S'$, and will call (S, S') a $\mathcal{U}_{\neq}$**-pair**.

It is clear that if we have $\mathcal{U}(S, S')$, then there exists a M-uple Q such that: $C(S') = C(S).Q$. It can be shown that Q and $out(w)$ commute. More precisely, $E(S) = E(S')$ and $S \leq_g S'$ and $out(w) \leq Q.out(w)$ (where Q satisfies $C(S') = C(S).Q$) is equivalent with $\mathcal{U}(S, S')$.

We have the following results:

Lemma 1 *Let S and S' be two global states of $\mathcal{S}$ such that $S \leq_g S'$ and $t \in T$ a single transition fireable in S. Then t is fireable from S' and the local states of the two reached global states are identical, i.e. if $S \xrightarrow{t} S_1$ and $S \leq_g S'$, then $S' \xrightarrow{t} S'_1$ and $E(S_1) = E(S'_1)$.*

Lemma 2 *Let $w = t_1 \dots t_n$ and $S, S' \in GS$ such that $S \xrightarrow{w}_* S'$ and $\mathcal{U}_{\neq}(S, S')$. Note S_1 the global state such that $S \xrightarrow{t_1} S_1$. Then $S' \xrightarrow{t_1} S'_1$ and $\mathcal{U}_{\neq}(S_1, S'_1)$.*

We have: $S_1 \xrightarrow{t_2} S_2 \dots S_{n-1} \xrightarrow{t_n} S' \xrightarrow{t_1} S'_1 \dots S'_{n-1} \xrightarrow{t_n} S''$, with, for all i: $\mathcal{U}_{\neq}(S_i, S'_i)$ and $\mathcal{U}_{\neq}(S', S'')$. We call the states $S_1 \dots S_{n-1}$ the **linking states** between S and S'.

Theorem 1 *Let S and S' be two global states of $\mathcal{S}$. If $S \xrightarrow{w}_* S'$ and $\mathcal{U}_{\neq}(S, S')$, then the transition system $\mathcal{S}$ is infinite.*

This is a direct consequence of lemma 0.2. It is possible to build an infinite sequence of global states, linked by the sequence of transitions $t_1 \dots t_n$. Hence, the infinite transition system $\mathcal{S}$ exhibits at least one partial regularity: the infinite sequence $(t_1 \dots t_n)^*$. The computation of the relation $\mathcal{U}_{\neq}$ can be made during the construction of the reachability graph corresponding to a Cfsm system (cf. Jéron and Jard (1993)).

We also need the following result, stating that when different pair of states are linked by $\mathcal{U}_{\neq}$ for a same sequence of transitions w, then the channel growth is the same for the two pairs.

Lemma 3 *Let S, S', V, V' be four states of GS, and $w = t_1 \dots t_n$ such that: $\mathcal{U}_{\neq}(S, S')$ and $\mathcal{U}_{\neq}(V, V')$ and $S \xrightarrow{w}_* S'$ and $V \xrightarrow{w}_* V'$. Let Q^S and Q^V be the two M-uples such that $C(S') = C(S).Q^S$ and $C(V') = C(V).Q^V$. Then: $Q^S = Q^V$.*

Proof. Omitted, cf. Quemener and Jéron (1996). □

Definition 4 *Let $\mathcal{S}$ be a transition system. Let $w \in T^*$ be a sequence of transitions, and $S_0, \dots, S_i, \dots$ an infinite sequence of global states of $\mathcal{S}$ such that, for all i, $S_i \xrightarrow{w}_* S_{i+1}$ and $\mathcal{U}_{\neq}(S_i, S_{i+1})$. Let Q be the M-uple such that, for all i, $C(S_{i+1}) = C(S_i).Q$ (Q exists and is unique according to 0.3). We say that $S_0, \dots, S_i, \dots$ is a $\mathcal{U}_{\neq}$-**chain**, which is* **linked by** *w, and whose* **growth** *is Q.*

We see in the next section how to finitely represent infinite graphs showing a complete regularity: they are composed of a finite number of patterns that repeat themselves, and can be represented with a graph grammar.

3 GRAPH GRAMMARS AND INFINITE REACHABILITY GRAPHS

We first give a more general definition of transition systems, that don't take into account the FIFO communication and initial states. We then show how graph grammars can represent such infinite labelled transition graphs.

Definition 5 *A* **labelled transition graph** *(in short, LTG) is a triple $\mathcal{S} = (St, Act, \rightarrow)$ where St is a set of* **states**, *Act is a finite alphabet of* **actions**, *and $\rightarrow \subseteq St \times Act \times St$ is the* **transition relation**.

We need some labels different of those of actions:

Definition 6 *A* **graded alphabet** *L is a finite set of letters such that with each letter $l \in L$ there is associated a positive integer called the* **arity** *of l. Thus L can be partitionned into sets L_i, with $l \in L_i$ if the arity of l is i.*

In the following, we fix a graded alphabet $L = \{l_1, \ldots, l_q\}$ with $L \cap Act = \emptyset$.

Definition 7 *An* hypergraph *G is a couple $(\mathcal{S}(G), H(G))$, where $\mathcal{S}(G) = (St_G, Act, \rightarrow_G)$ is a finite LTG and $H(G)$ is a finite set of* hyperarcs *H. An hyperarc H is denoted by $ls_1 \ldots s_n$. Its* label *is $l \in L_n$, and the s_i are n distinct states of St_G. We say that the states $s_1 \ldots s_n$ are* linked *by H.*

Definition 8 *A* deterministic graph grammar on *L (in short graph grammar) $\mathcal{G}$ consists of*

- *q hyperarc replacement rules $l_i x_1 \ldots x_{n_i} \triangleright G_i$ where l_i is of arity n_i, G_i is an hypergraph, and the states $x_1 \ldots x_{n_i}$ are distinct states of G_i; and*
- *an initial hypergraph G_0.*

The states $x_1 \ldots x_{n_i}$ are called *gluing states.*

Definition 9 *Given a graph grammar $\mathcal{G}$, and an hypergraph M, M* rewrites in one step *to an hypergraph N, and we note $M \rightarrow_{\mathcal{G}} N$, if, for some rule $(lx_1 \ldots x_n \triangleright G) \in \mathcal{G}$, we have:*

- *$\exists (z_1, \ldots, z_n) \in (St_M)^n : H = lz_1 \ldots z_n \in H(M)$, and*
- *for some matching function f^H, called the rewriting function associated with H, mapping x_i to z_i, and mapping injectively the other states of St_G to states outside of St_M:*

  - $St_N = St_M \cup \{f^H(s) | s \in St_G\}$
  - $\rightarrow_N = \rightarrow_M \cup \{(f^H(s), a, f^H(t)) | s \stackrel{a}{\rightarrow}_G t\}$
  - $H(N) = H(M) \backslash \{lz_1 \ldots z_n\} \cup \{lf^H(s_1) \ldots f^H(s_n) | ls_1 \ldots s_n \in H(G)\}$

Intuitively, a rewriting step can be described as follows: you first take back the hyperarc you want to rewrite, and then you glue the hypergraph G, along the gluing states, on the states linked by the hypergraph you have rewritten.

Note that, in general, $\rightarrow_{\mathcal{G}}$ is not a functional relation, even though $\mathcal{G}$ is deterministic. Nevertheless, if we let $M \rightarrow_{\mathcal{G},H} N$ denote the rewriting of an hyperarc H, then:

$$M \rightarrow_{\mathcal{G},H_1} \ldots \rightarrow_{\mathcal{G},H_n} N \text{ iff } M \rightarrow_{\mathcal{G},H_{\pi(1)}} \ldots \rightarrow_{\mathcal{G},H_{\pi(n)}} N$$

for any $H_i \in H(M)$, and for any permutation π on $\{1, \ldots, n\}$. Thus, it makes sense to define a *complete parallel rewriting step* $M \Rightarrow_{\mathcal{G}} N$ as follows:

$$M \Rightarrow_{\mathcal{G}} N \text{ if } M \rightarrow_{\mathcal{G},H_1} \ldots \rightarrow_{\mathcal{G},H_n} N,$$

where $H(M) = \{H_1, \ldots, H_n\}$. On that basis, we define $\mathcal{G}^\omega(M)$, the set of infinite graphs generated from the axiom M according to the deterministic graph grammar $\mathcal{G}$ as follows.

Definition 10 *$\mathcal{G}^\omega(M)$ is the set of graphs N for which there exists an infinite sequence of hypergraphs $(\mathcal{G}^n(M))_{n\geq 0}$, such that $\mathcal{G}^0(M) = M$, and for all n, $\mathcal{G}^n(M) \Rightarrow_{\mathcal{G}} \mathcal{G}^{n+1}(M)$, with limit $N = (St_N, Act, \rightarrow_N)$ where:*

- $St_N = \{s | \exists i, s \in St_{\mathcal{G}^i(M)}\}$
- $\rightarrow_N = \{(s,a,t) | \exists i, s \stackrel{a}{\rightarrow}_{\mathcal{G}^i(M)} t\}$

Since $\mathcal{G}$ is deterministic, $\mathcal{G}^\omega(M)$ has a single element up to graph isomorphism called the infinite graph generated from M by $\mathcal{G}$.

The axiom hypergraph G_0 being given, the graph grammar $\mathcal{G}$ defines a unique infinite LTG $\mathcal{G}^\omega(G_0)$.

Figure 0.3 shows an example of a graph grammar $\mathcal{G}$, with two rewriting rules, such that $\mathcal{G}^\omega(G_0)$ is isomorphic to the reachability graph of Fig. 0.2. The hyperarcs of label l1 have an arity of 3, and those of label l2 have an arity of 1. Only the labels of gluing states are represented. Of course, the states of the hypergraphs of $\mathcal{G}$ can be labelled with properties that will be true for all the states of the infinite graph that they represent. For example, they can be labelled with the corresponding local states, or the prefixes of the channel contents.

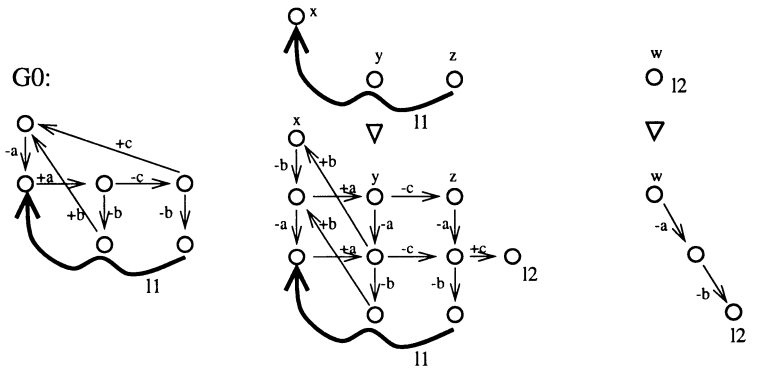

Figure 3 A graph grammar generating the infinite graph of Figure 0.2.

4 REPEATING SEQUENCES

Roughly speaking, an infinite graph can be represented with a graph grammar if it is composed of a finite number of patterns that are regularly repeated. The relation $\mathcal{U}_{\neq}$ presented above enables us to detect what we will call a "skeleton" of a pattern, defined by the sequences of transitions that can be infinitely repeated. But, what about the

transitions issued from the states of a "skeleton", and that form what we could call, following our analogy, the "body" of a pattern? In general, those transitions are not the same. Hence, even if the "skeleton" is the same, it is not sure that there is a common pattern "body". This is that problem we are interested in: to prove that a pattern "body" is also infinitely repeated, along with its corresponding skeleton. For that, we will show that if a "body" is repeated twice, it is infinitely repeated.

Lemma 4 *Let $S_0^0, \ldots, S_n^0, \ldots$ be a $\mathcal{U}_{\neq}$-chain, linked by w, of growth Q. Let $t_1, \ldots, t_{p+1}$ be $p+1$ transitions such that:*

$$\forall i : \exists (S_i^1, \ldots, S_i^p) \in GS^p \text{ such that } \forall j \in [0..p-1], S_i^j \overset{t_{j+1}}{\rightarrow} S_i^{j+1}$$

and t_{p+1} is fireable in S_0^p (resp. S_1^p), leading to a state S_0^{p+1} (resp. S_1^{p+1}).
Then, for all i greater or equal than 2, t_{p+1} is fireable in S_i^p.

Proof. First, we recall that: $\forall i, C(S_{i+1}^0) = C(S_i^0).Q = C(S_0^0).Q^{i+1}$.

We prove the lemma by induction on p. For $p = 0$, since t_1 is fireable in S_0^0, we have: $in(t_1) \leq C(S_0^0)$. Hence, $in(t_1) \leq C(S_0^0).Q^i = C(S_i^0)$. Thus, t_1 is fireable in S_i^0.

We now separate the proof in three cases. The induction hypothesis will be used for expressing channel contents in case 3.

Case 1: $in(t_1 \ldots t_p t_{p+1}) \leq C(S_0^0)$

Then, $in(t_1 \ldots t_{p+1}) \leq C(S_0^0).Q^i = C(S_i^0)$. Hence, the complete sequence $t_1 \ldots t_{p+1}$ is fireable from S_i^0, and, especially, t_{p+1} is fireable from S_i^p.

Case 2: $(in(t_1 \ldots t_p) \leq C(S_0^0)) \wedge (C(S_0^0) < in(t_1 \ldots t_{p+1}))$

Then, there exists a M-uple Z such that: $C(S_0^0) = in(t_1 \ldots t_p).Z$, another M-uple Z' such that: $in(t_1 \ldots t_{p+1}) = C(S_0^0).Z'$. We have: $in(t_{p+1}) = Z.Z'$.

Since $t_1 \ldots t_p$ is fireable in S_i^0, we have: $in(t_1 \ldots t_p).C(S_i^p) = C(S_0^0).Q^i.out(t_1 \ldots t_p)$. This gives: $C(S_i^p) = Z.Q^i.out(t_1 \ldots t_p)$.

Since t_{p+1} is fireable in S_0^p and S_1^p, we have:

$$\begin{array}{llll} C(S_0^p) & = Z.out(t_1 \ldots t_p) & \geq in(t_{p+1}) & = Z.Z' \\ C(S_1^p) & = Z.Q.out(t_1 \ldots t_p) & \geq in(t_{p+1}) & = Z.Z' \end{array}$$

Hence, there exists a couple of M-uples (T, U) such that $out(t_1 \ldots t_p) = Z'.T$ and $Q.out(t_1 \ldots t_p) = Z'.U$, i.e. such that: $Q.Z'.T = Z'.U$. Or, for each j-th component of those M-uples: $Q_j.Z'_j.T_j = Z'_j.U_j$.

Let us study that relation. If we suppose that the length of Z'_j is lesser or equal than the length of Q_j, and by renaming P_j the word Z'_j, there exists a word R_j such that: $Z'_j = P_j$ and $Q_j = P_j.R_j$. If we suppose that the length of Z'_j is strictly greater than the length of Q_j, there exists a word Z''_j such that: $Z'_j = Q_j.Z''_j$. We can then write: $Q_j^2.Z''_j.T_j = Q_j.Z''_j.U_j \iff Q_j.Z''_j.T_j = Z''_j.U_j$. Hence, Z''_j satisfies the same relation than Z'_j. We can iterate that procedure n_j times, until we obtain a component $Z_j^{(n_j)}$ whose length is lesser than that of Q_j. We then note P_j that final component.

Overall, we obtain: $\forall j : \exists n_j, P_j, R_j$ such that $Z'_j = Q_j^{n_j}.P_j$ and $Q_j = P_j.R_j$.

Hence, for all j:

$$Q_j.Z'_j = P_j.R_j.\overbrace{P_j.R_j.\ldots.P_j.R_j}^{n_j \text{ times}}.P_j = \overbrace{P_j.R_j.\ldots.P_j.R_j}^{n_j \text{ times}}.P_j.R_j.P_j = Z'_j.R_j.P_j$$

Thus, $Q.Z' = Z'.R.P$, where P and R are the M-uples built with the components P_j and R_j, which implies: $Z' \leq Q.Z'$. This gives us immediately: $Z' \leq Q^n.Z'$, for all n greater than 1.

And we have seen previously: $Z' \leq Q.out(t_1 \ldots t_p)$. Hence:

$$\begin{array}{lll} \forall i, i \geq 2: & Q^{i-1}.Z' & \leq Q^i.out(t_1 \ldots t_p) \\ & Z' & \leq Q^i.out(t_1 \ldots t_p) \\ & Z.Z' & \leq Z.Q^i.out(t_1 \ldots t_p) \\ & in(t_{p+1}) & \leq C(S_i^p) \end{array}$$

That is, t_{p+1} is fireable in S_i^p, and leads to a state S_i^{p+1}. We now determine $C(S_i^{p+1})$. First, we have:

$$\begin{array}{rcl} in(t_{p+1}).C(S_i^{p+1}) & = & C(S_i^p).out(t_{p+1}) \\ Z.Z'.C(S_i^{p+1}) & = & Z.Q^i.out(t_1 \ldots t_p).out(t_{p+1}) \\ Z'.C(S_i^{p+1}) & = & Q^i.Z'.T.out(t_{p+1}) \end{array}$$

And we know: $Q.Z' = Z'.R.P$. Hence, $Q^i.Z' = Z'.(R.P)^i$. Finally, by noting V the M-uple $R.P$, we obtain: $C(S_i^{p+1}) = V^i.T.out(t_{p+1})$.

Case 3: $(C(S_0^0) < in(t_1 \ldots t_p)) \wedge (\exists (X, Y) : \forall i, C(S_i^p) = X^i.Y.out(t_p))$

We make here an assumption about the contents of channels. As we have seen in case 2, that assumption is satisfied the first time that the sequence $t_1 \ldots t_p$ is such that $C(S_0^0) < in(t_1 \ldots t_p)$. We will verify that this condition is satisfied for further steps at the end of that case.

First, t_{p+1} is fireable in S_0^p and S_1^p. Hence: $Y.out(t_p) \leq in(t_{p+1})$ and $X.Y.out(t_p) \leq in(t_{p+1})$. This implies that there exists two M-uples W and W' such that: $Y.out(t_p) = in(t_{p+1}).W$ and $X.Y.out(t_p) = in(t_{p+1}).W'$. We thus have: $X.in(t_{p+1}).W = in(t_{p+1}).W'$.

Hence, there exists a M-uple V such that: $W' = V.W$ and $X.in(t_{p+1}) = in(t_{p+1}).V$.

We thus have, for all i: $X^i.Y.out(t_p) = X^i.in(t_{p+1}).W = in(t_{p+1}).V^i.W$. That is: $in(t_{p+1}) \leq X^i.Y.out(t_p) = C(S_i^p)$.

Hence, t_{p+1} is fireable in S_i^p. We now determine $C(S_i^{p+1})$:

$$\begin{array}{rcl} in(t_{p+1}).C(S_i^{p+1}) & = & C(S_i^p).out(t_{p+1}) \\ in(t_{p+1}).C(S_i^{p+1}) & = & X^i.Y.out(t_p).out(t_{p+1}) \\ in(t_{p+1}).C(S_i^{p+1}) & = & in(t_{p+1}).V^i.W.out(t_{p+1}) \\ C(S_i^{p+1}) & = & V^i.W.out(t_{p+1}) \end{array}$$

The assumption about channel contents is satisfied for the next step of induction, with, of course, different M-uples. □

Corollary 1 *Let $S_0^0, \ldots, S_n^0, \ldots$ be a $\mathcal{U}_{\neq}$-chain, linked by w, of growth Q. Let $t_1, \ldots, t_p$ be p transitions such that:*

$$\forall \delta, \delta = 0 \text{ or } \delta = 1 : \exists (S_\delta^1, \ldots, S_\delta^p) \in GS^p \text{ such that } \forall j \in [0..p-1], S_\delta^j \xrightarrow{t_{j+1}} S_\delta^{j+1}$$

Then, for all i greater or equal than 2:

$$\exists (S_i^1, \ldots, S_i^p) \in GS^p \text{ such that } \forall j \in [0..p-1], S_i^j \xrightarrow{t_{j+1}} S_i^{j+1}.$$

And: $E(S_i^p) = E(S_0^p) = E(S_1^p)$;
And:

$$\begin{cases} \exists Z, & C(S_i^p) = Z.Q^i.out(t_1 \ldots t_p) & \text{if } in(t_1 \ldots t_p) \leq C(S_0^0) \\ \exists (X, Y), & C(S_i^p) = X^i.Y.out(t_p) & \text{if } in(t_1 \ldots t_p) > C(S_0^0) \end{cases}$$

where X, Y, and Z are M-uples determined by $C(S_0^0), C(S_1^0)$, and the transitions $t_1 \ldots t_p$.

Proof. Omitted, cf. Quemener and Jéron 1996. Direct consequence of lemma 0.4. □

We have shown that when a sequence is fireable from two states linked by $\mathcal{U}_{\neq}$, then that sequence is fireable from all the following states linked by $\mathcal{U}_{\neq}$. It remains to show that two such sequences remain connected in the same way when they are infinitely repeated.

Lemma 5 *Let $S_0, \ldots, S_i, \ldots$ and $S'_0, \ldots, S'_i, \ldots$ be two $\mathcal{U}_{\neq}$-chains, linked respectively by w and w', of respective growths Q and Q'. Let $t_1, \ldots, t_p$ be a sequence of transitions fireable from S_0 (resp. S_1), leading to U_0 (resp. U_1); and $t'_1, \ldots, t'_q$ be a sequence of transitions fireable from S'_0 (resp S'_1), leading to U'_0 (resp U'_1), such that $U_0 = U'_0$ and $U_1 = U'_1$. We know (cf. 0.1) that, for all i, $t_1 \ldots t_p$ (resp. $t'_1 \ldots t'_q$) is fireable from S_i (resp. S'_i), leading to U_i (resp. U'_i). We have, for all i: $U_i = U'_i$.*

Proof. Omitted, cf. Quemener and Jéron 1996. □

We now show how patterns can be attached to patterns in the same way than the example given above. More precisely, we show that, if a repeatable sequence (detected by a $\mathcal{U}_{\neq}$ relation) is fireable from the two first states of a sequence of states linked by $\mathcal{U}_{\neq}$, then the $\mathcal{U}_{\neq}$ relation is repeated for all following patterns. Hence, the results of 0.4 can also be applied to those new patterns.

Lemma 6 *Let $S_0, \ldots, S_i, \ldots$ be a $\mathcal{U}_{\neq}$-chain, linked by w, of growth Q. Let $t = t_1 \ldots t_p$ and $v = v_1 \ldots v_q$ be two words such that tv is fireable from S_0 and S_1; $S_0 \xrightarrow{t}_* U_0 \xrightarrow{v}_* W_0$ and $S_1 \xrightarrow{t}_* U_1 \xrightarrow{v}_* W_1$; $\mathcal{U}_{\neq}(U_0, W_0)$ and $\mathcal{U}_{\neq}(U_1, W_1)$. We know (cf. 0.1) that, for all i, tv is fireable in S_i, and $S_i \xrightarrow{t}_* U_i \xrightarrow{v}_* W_i$. We have, for all i: $\mathcal{U}_{\neq}(U_i, W_i)$.*

Proof. Omitted, cf. Quemener and Jéron (1996). □

5 THE PROPOSED ALGORITHM

5.1 Overview of the Algorithm

Let fix now a Cfsm system, and let $\mathcal{S}$ be the transition system modeling its behaviour. Following Jéron and Jard (1993), we can test during the construction of $\mathcal{S}$ whether a finite sequence of transitions will be infinitely repeated. If the algorithm detects pairs of states S and S' such that $\mathcal{U}_{\neq}(S, S')$, it doesn't fire the transitions issued from S'. In that way, a **reduced transition system**, that we note $Red_0(\mathcal{S})$ is built. The algorithm can not ensure that it will detect all infinite transition systems: a non-regular growth of channel contents can not be detected.

That is, the algorithm proposed by Jéron and Jard (1993) can deliver three kinds of results:

- the algorithm stops, and no pair of states (S, S') such that $\mathcal{U}_{\neq}(S, S')$ is detected: then, $\mathcal{S}$ is finite; or,
- the algorithm stops, and some pairs of states (S, S') such that $\mathcal{U}_{\neq}(S, S')$ are detected: then, $\mathcal{S}$ is infinite, and we know a part of it (the infinitely repeating sequences of transitions); or,
- the algorithm does not stop: we can not conclude; maybe $\mathcal{S}$ is finite but too big for being generated, or infinite but has not been detected.

We will suppose that we are in the second situation, and will try to use the results presented in section 4 for determining completely $\mathcal{S}$ and represent it with a graph grammar. Our algorithm is not guaranteed to terminate, nor to provide a graph grammar representing $\mathcal{S}$ for all transition systems that could be represented in that way. We only guarantee that if our algorithm delivers a graph grammar $\mathcal{G}$, then $\mathcal{G}^{\omega}(G_0)$ is isomorphic to $\mathcal{S}$. Hence, $\mathcal{G}$ finitely represents $\mathcal{S}$.

The graph grammars obtained will be of a special form. They will consist of an hypergraph G_0 and q rewriting rules $l_i x_1 \dots x_{n_i} \rhd G_i$ where:

- $H(G_0)$ is composed of exactly one hyperarc, labelled with l_1; and,
- $H(G_i)$, for all $0 < i < q$, is composed of exactly two hyperarcs, one labelled with l_i and the other with l_{i+1}; and,
- $H(G_q)$ is composed of exactly one hyperarc, labelled with l_q.

The graph grammar of Fig. 0.3 follows those conditions. The general form of the infinite transition systems represented by such graph grammars can be described as follows: an initial graph calls a pattern (G_1) that will call itself infinitely, and eventually an other pattern (G_2) that will call itself infinitely, and eventually another pattern, until the pattern G_q that only calls itself. How will we detect those patterns?

We start with the reduced transition system $Red_0(\mathcal{S})$. Let n_1 be the number of pairs (S_i, S_i') such that we have $\mathcal{U}_{\neq}(S_i, S_i')$, and $w_i = t_i^1 \dots t_i^m$ the word such that $S_i \stackrel{w_i}{\rightarrow}_* S_i'$. Our algorithm will work only if there are no common states between the pairs (S_i, S_i') and (S_j, S_j') and the linking states between them for $i \neq j$, and we further suppose that all the transitions fireable from S_i' go to states in $Red_0(\mathcal{S})$, except for t_i^1.

Next, we copy the n_1 states S'_i and continue from them the building of $\mathcal{S}$, allowing only t_i^1 as fireable transition from S'_i. We will call the partial transition system eventually obtained $Red_1(\mathcal{S})$. We already know a part of that construction: the sequence $w_i = t_i^1 \dots t_i^m$ leads from S'_i to S''_i, and we have $\mathcal{U}_{\neq}(S'_i, S''_i)$. In the same way that for the building of $Red_0(\mathcal{S})$, we don't fire the transitions t_i^1 from the states S''_i, neither do we fire transitions eventually issued from n_2 other states T'_i, such that there exists T_i, and we have $\mathcal{U}_{\neq}(T_i, T'_i)$. If we detect transitions leading to states in $Red_0(\mathcal{S})$ other than the states S'_i, this means that the states S'_i don't form a correct boundary between the initial graph and the first pattern: it provokes a failure. We must also test whether the conditions imposed in $Red_0(\mathcal{S})$ for the pairs (S_i, S'_i) are satisfied for the pairs (T_i, T'_i). If this is the case, $Red_1(\mathcal{S})$ is a candidate for being the first pattern G_1 of our graph grammar (and $Red_0(\mathcal{S})$ will of course be G_0).

We have now to test the putative pattern $Red_1(\mathcal{S})$. It can be simply made by applying the same construction to states S''_i, that the construction which gave $Red_1(\mathcal{S})$ from states S'_i. If the result is isomorphic to $Red_1(\mathcal{S})$, and $\mathcal{U}_{\neq}$-pairs similar to the pairs (T_i, T'_i) are detected, we know, according to the results of section 3, that $Red_1(\mathcal{S})$ will be infinitely repeated, and we can begin the construction of $Red_2(\mathcal{S})$ by using the states T'_i. If this is not the case, we can glue $Red_0(\mathcal{S})$ and $Red_1(\mathcal{S})$ along the states S'_i. The new $Red_0(\mathcal{S})$ obtained in that way satisfies the conditions imposed to $Red_0(\mathcal{S})$ but its set of boundary states is now composed of the states S''_i and T'_i that will be used to build a new $Red_1(\mathcal{S})$.

If the testing of $Red_1(\mathcal{S})$ has been succesful, we can apply the same construction to states T'_i, and create $Red_2(\mathcal{S})$. Those successive buildings stop when there is no other $\mathcal{U}_{\neq}$-pairs in $Red_q(\mathcal{S})$ than the initial ones. If the repetition test is also fulfilled for $Red_q(\mathcal{S})$, we have then obtained a correct pattern G_q.

5.2 Sketch of the Algorithm

We now give a more precise outline of our algorithm in Fig. 0.4. Starting with a Cfsm system, it is not guaranteed to stop (the building of reduced transition systems $Red_i(\mathcal{S})$ is not guaranteed to terminate). If it stops, either it gives a `FAIL` verdict, or produces q patterns G_i that will compose our graph grammar. Within each pattern G_i, some sets of distinguished states are defined:

- the states (ex_i^{i+1}) are those states of G_i (for $0 \leq i < q$) that will be linked by the hyperarc of label l_{i+1};
- the states (out_i) are those states of G_i (for $0 < i \leq q$) that will be linked by the hyperarc of label l_i;
- and the states (in_i) are the gluing states of G_i (for $0 < i \leq Q$).

The states (ex_i^{i+1}) and (out_i) are determined during the construction of $Red_i(\mathcal{S})$: they are the states in which a $\mathcal{U}_{\neq}$ relation can be detected. The states (in_i) are exactly the states (ex_{i-1}^i).

The procedure **representing** begins with building $Red_0(\mathcal{S})$. If the conditions depicted in 0.5.1 are fulfilled, $Red_0(\mathcal{S})$ can be used as initial graph G_0. The procedure **new_pattern** is then called, for determining G_1. That procedure has three arguments: a set of states to which states of the new pattern can not be directly linked (Forbidden_States); a set of states that are used as starting point for the new pattern (Beginning_States); and the new pattern number (p_n). The procedure **new_pattern** builds $Red_{p\_n}(\mathcal{S})$, verify

procedure new_pattern(Forbidden_States; Beginning_States; p_n)
BEGIN
Build $Red_{p\_n}(\mathcal{S})$.
IF $Red_{p\_n}(\mathcal{S}) \cap Forbidden\_States \neq \emptyset$ THEN FAIL;
IF not conditions_fulfilled($Red_{p\_n}(\mathcal{S})$) THEN FAIL;
IF not repeat_pattern($Red_{p\_n}(\mathcal{S})$)
THEN $G_{p\_n-1} := G_{p\_n-1} \oplus Red_{p\_n}(\mathcal{S})$;
$(ex^{p\_n}_{p\_n-1}) := (out_{p\_n-1}) \cup (ex^{p\_n}_{p\_n-1})$, $(in_{p\_n}) := (ex^{p\_n}_{p\_n-1})$;
new_pattern(Forbidden_States $\cup(Red_{p\_n}(\mathcal{S})\backslash(ex^{p\_n}_{p\_n-1}))$; (in_{p_n}); p_n);
END **new_pattern**;
ENDIF;
$G_{p\_n} := Red_{p\_n}(\mathcal{S})$, $(in_{p\_n+1}) := (ex^{p\_n+1}_{p\_n})$;
IF $(in_{p\_n+1}) \neq \emptyset$
THEN new_pattern(Forbidden_States $\cup(G_{p\_n}\backslash(ex^{p\_n+1}_{p\_n}))$; (in_{p_n+1}); p_n +1);
END **new_pattern**;

procedure representing(Cfsm_system)
BEGIN
Build $Red_0(\mathcal{S})$.
IF conditions_fulfilled($Red_0(\mathcal{S})$)
THEN $G_0 := Red_0(\mathcal{S})$, $(in_1) := (ex^1_0)$;
new_ pattern($G_0\backslash(ex^1_0)$; (in_1); 1);
ELSE FAIL;
ENDIF;
Linking all states (ex^i_{i-1}) and (out_i) by an hyperarc of label l_i.
RETURN $\mathcal{G} = (G_0, (l_i in^1_i \ldots in^{n_i}_i \rhd G_i)^{i=q}_{i=1})$.
END **representing**;

Figure 4 The algorithm for representing some Cfsm systems with a graph grammar.

that the necessary conditions are fulfilled, and that the states of $Red_{p\_n}(\mathcal{S})$ can't reach directly the Forbidden_States *. If this is true, it can be tested whether the new pattern is effectively repeated. If this is not the case, $Red_{p\_n}(\mathcal{S})$ is glued with G_{p_n-1} (symbolized by the $\oplus$ operator), and **new_pattern** is recalled for the same pattern number. If the pattern is repeated, then there will be no other transitions fireable from the following patterns than those of the first two patterns, since the channel contents of the states of the following patterns have the same prefixes than those of the first two patterns. Hence, we have determined G_{p_n}, and we eventually recall **new_pattern** for the following pattern number.

Theorem 2 *If the procedure* **representing** *terminates, and returns a graph grammar* $\mathcal{G}$*, then the infinite graph* $\mathcal{G}^\omega(G_0)$ *is isomorphic to the infinite transition system* $\mathcal{S}$ *defined by the Cfsm system argument of* **representing**.

*They can reach those states **via** the states (in_{p_n}) that are the same than the states $(ex^{p\_n}_{p\_n-1})$.

Proof. (Sketch) Let S be a state of $\mathcal{G}^{\omega}(G_0)$. Then, the sequence of transitions leading to S from the initial state S_0 in G_0 is built with various sequences found in the hypergraphs G_i. Those sequences are sequences of $\mathcal{S}$ (obtained during the constructions of the $Red_i(\mathcal{S})$) that have been tested as being repeating sequences (in the sense of Section 4). Hence, they can be repeated in the same way in $\mathcal{S}$, and S is a global state of $\mathcal{S}$.

Let S be a global state of $\mathcal{S}$. If S is a state of $Red_0(\mathcal{S})$, then S is a state of G_0, and hence of $\mathcal{G}^{\omega}(G_0)$. If not, the sequence leading from S_0 to S goes necessarily through one of the states (ex_0^1), leaving it by the first transition of the corresponding repeated sequence. Since no more transitions are fireable from the following patterns when the repeating test is satisfied, we know that all transitions of $\mathcal{S}$ are represented in the hypergraphs G_i, and hence that S is a state of $\mathcal{G}^{\omega}(G_0)$.

The states of $\mathcal{S}$ and $\mathcal{G}^{\omega}(G_0)$ are the same, and are connected in the same way because of lemma 0.5. □

6 CONCLUSION

We have proposed an algorithm enabling to finitely represent some infinite reachability graphs of Cfsm systems with graph grammars. Those representations can be used for solving the model-checking problem of the alternation-free μ-calculus (a branching-time temporal logic including CTL) on the corresponding infinite reachability graphs, as has been shown by Burkart and Quemener (1996). Hence, we propose here a possibility for verifying properties of infinite-state systems defined by systems of finite-state machines communicating with FIFO channels. We are not aware of other results in that field of research, except the work of Boigelot and Godefroid (1996).

The undecidability of the general problem implies that our algorithm can not produce a graph grammar for all infinite transition systems that could be represented in such a way. So, intensive testing is necessary to decide the usefulness and range of that new verification method. Nevertheless, we are quite optimistic in that regard. Indeed, infinity of the reachability graph of communication protocols is often caused by the possibility of unbounded emissions, in the same way as in our example. Our algorithm should then succeed in deriving a graph grammar.

REFERENCES

Baeten, J.C.M., Bergstra, J.A. and Klop, J.W. (1987) Decidability of bisimulation equivalence for processes generating context-free languages. In *PARLE'87*, LNCS 259, pages 94-113. Springer.

Boigelot, B. and Godefroid, B. (1996) Symbolic verification of communication protocols with infinite state spaces using QDDs. In *CAV'96*, to appear in LNCS.

Bochmann, G. (1978) Finite state descriptions of communication protocols. *Computer Networks*, **2**.

Burkart, O. and Quemener, Y.-M. (1996) Model-checking of infinite graphs defined by graph grammars. Publication interne 995, IRISA. Presented at *Infinity'96*. Accessible at `http://www.irisa.fr/EXTERNE/bibli/pi/PI-995.html`.

Burkart, O. and Steffen, B. (1992) Model-checking for context-free processes. In *CONCUR'92*, LNCS 836, pages 123-137. Springer.

Brand, D. and Zafiropoulo, P. (1983) On communicating finite-state machines. *Journal of the ACM*, **2**, pages 323-342.

Caucal, D. (1992) On the regular structure of prefix rewriting. *Theoretical Computer Science*, **106**, pages 61-86.

Courcelle, B. (1990) Graph rewriting: an algebraic and logic approach. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, chapter 5, pages 193-242. Elsevier Science Publisher B.V.

ISO 9074 (1989) *Estelle: a formal description technique based on an extended state transition model.* ISO TC97/SC21/WG6.1.

Hüttel, H. and Stirling, C. (1991) Actions speak louder than words: proving bisimilarity for context-free processes. In *6th Annual IEEE Symposium on Logic in Computer Science*, pages 376-386. IEEE Computer Society Press.

Hungar, H. and Steffen, B. (1993) Local model-checking for context-free processes. In *ICALP'93*, LNCS 700, pages 593-605. Springer.

Jéron, T. and Jard, C. (1993) Testing for unboundedness of FIFO channels. *Theoretical Computer Science*, **113**, pages 93-117.

Muller, D.E. and Schupp, P.E. The theory of ends, pushdown automata, and second-order logic. *Theoretical Computer Science*, **37**, pages 51-75.

Purushothaman Iyer, S. (1993) A note on model-checking context-free processes. In *North American Process Algebra Workshop.*

Quemener, Y.-M. and Jéron, T. (1995) Model-checking of infinite Kripke structures defined by graph grammars. In Corradini, A. and Montanari, U., editors, *Segragra'95*, ENTCS 2, pages 67-74. Elsevier Science B.V.

Quemener, Y.-M. and Jéron, T. (1996) Finitely representing infinite reachability graphs of CFSMs with graph grammars. Publication interne 994, IRISA. Long version of current paper. Accessible at: `http://www.irisa.fr/EXTERNE/bibli/pi/PI-994.html`.

CCITT Recommendation Z.100. (1988) *Specification and description language SDL.* Blue Book, Vol. X.1-X.5.