Testing for Unboundedness of Fifo Channels *

Thierry JERON, Claude JARD
IRISA, Campus de Beaulieu,
35042 RENNES Cedex, FRANCE
e-mail: jeron@irisa.fr, jard@irisa.fr

Abstract

Undecidability of the unboundedness problem for specification models allowing fifo channels was proved a few years ago by Brand and Zafiropulo. The paper investigates a testing approach of that problem. Dealing with the general framework of systems communicating through fifo channels, we find a sufficient condition for unboundedness based on a relation between the nodes of the reachability tree. The construction of the resulting reduced tree can then be applied as well to communicating finite state machines as to Fifo-Nets. Moreover, the test extends existing decidability results. As a matter of fact, it becomes a decision procedure for a class of systems strictly including linear and monogeneous systems which are the two essential classes in which decidability is already known. In order to conclude our study on a practical view, we show that a few modifications of the relation make the test available for Estelle specifications.

1 Introduction

This paper deals with the analysis of programs involving fifo queues or channels. The fifo channel object is widely used in computer science and is used as a temporary structured method. The most spread example is given by the specification of distributed software. Programs are then described as a set of processes and communication memories. In many cases, fifo channels are used in order to model the asynchronous hardware technology which is the most common one.

The main specification models explicitly describing fifo channels are communicating finite state machines (Cfsm) [Boc78], Fifo-Nets [MM81] which extend Petri nets and some specification languages like Estelle [90789]. Estelle is a Formal Description Technique now normalized by ISO, which basic model is communicating finite state machines augmented with Pascal constructs.

The whole behaviour of such a specification is modelled by a product of transition systems in which all interleavings of local actions are considered as possible computations. Thus, if fifo channels are not bounded *a priori*, the behaviour of such protocols

*A French version of this paper constitutes a chapter of the Ph. D Thesis of Thierry Jéron [Je91a]. A short version was presented in STACS'91 [Je91b] specifications can lead to channels overflows. But the computation of bounds, which is a wished result of a verification, is often impossible to perform.

Knowing if a channel is bounded or not is an interesting result at least for two reasons:

- in order to choose the suitable verification technique,
- and it is a property to provide to the programmer who will have for example to plan a flow control mechanism.

In the past, there has been a few papers written on that subject [GGLR87, CF87, Fin86, Fin88]. We recall the main ones here. In most of them, other reachability problems were approached and, as these problems are generally undecidable, authors restrict to sub-classes in which some of the problems are decidable.

In our paper we only deal with the unboundedness problem and we adopt a quite different point of view which is a more practical one and which surprisingly permits to improve known theoretical results. Therefore, our contribution is two-fold:

- Practical contribution: we describe a test for unboundedness which can be applied to all programs in the general class that we consider.
- Theoretical contribution: we prove that this test is a decision procedure for unboundedness in a sub-class of programs which strictly includes the main ones in which decidability was already proved.

The paper is organized as follows. Section 2 begins with a few notations about combinatorics on words, generalized to p-tuples of words. After recalling a few definitions on labelled transition systems we give a formal definition of communicating finite state machines and Fifo-Nets. We then define a general model of transition systems that we call Transition Systems with Fifo Channels (TSFC) which can express the behaviours of most specification models allowing fifo channels. We then introduce the unboundedness problem in that framework and recall the main decidability results for Cfsm systems and Fifo-Nets. These results allow us to argue out of the testing approach. Section 3 introduces a new relation between nodes of reachability trees of TSFC. By a close examination of the behaviour of channels contents, we prove that this relation provides a sufficient condition for unboundedness. The induced test, based on the construction of a reduced tree, is then a decision procedure for a class of systems strictly including linear and monogeneous systems. Those classes were almost the only not obvious ones in which unboundedness was proved decidable [GGLR87, CF87, Fin86, Fin88]. Section 4 describes an algorithm implementing this test and gives some applications to Cfsm systems. We conclude with some indications upon how this test can be applied to Estelle specifications.

2 The unboundedness problem

2.1 Notations and definitions

2.1.1 Combinatorics on words

Let us begin with a few classical notations and definitions about combinatorics on words. For readability reasons, they are extended to p-tuples of words.

Let A be a finite *alphabet*, the elements of which are called *letters*. A finite sequence of *letters* is called a *word* and is denoted

$$x = a_1 a_2 \dots a_n$$

The empty word is ϵ.

The sets A^* and A^+ respectively denote the set of all words over A and the set of all non-empty words of A^* ($A^+ = A^* - \{\epsilon\}$).

The set of p-tuples of words over the alphabets $A_1, \dots, A_p$ is the cartesian product

$$\times_{i=1}^{p} A_i^* = A_1^* \times A_2^* \times \dots \times A_p^*$$

A p-tuple of words X will be written

$$X = < x_1, x_2, \dots, x_p >$$

As usual, the *concatenation* of two words is denoted by a dot.

We also define the concatenation of two p-tuples of words as the concatenation of each coordinate:

$$\begin{array}{l} \text{let } X = < x_1, \dots, x_p > \text{ and } Y = < y_1, \dots, y_p > \in \times_{i=1}^{p} A_i^*, \\ \text{define } X.Y = < x_1.y_1, \dots, x_p.y_p > \end{array}$$

If $z = x.y$ we say that x (resp. y) is a *left (resp. right) factor* of z. $LF(z)$ and $RF(z)$ are the sets of left factors and right factors of z. The left factor usually defines a partial ordering $\leq$ between words called the *prefix ordering*:

$$x \leq z \Leftrightarrow \exists y \text{ such that } z = x.y$$

This is immediately generalized to p-tuples of words:

$$< x_1, \dots, x_p > \quad \leq \quad < z_1, \dots z_p > \quad \Longleftrightarrow \quad \forall i \in [1..p],\ x_i \leq z_i$$

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

Two words x and y *commute* if and only if $x.y = y.x$. Similarly, two p-tuples of words commute if and only if each pair of their coordinates commutes.

Remark: two non-empty words commute iff they are powers of the same word [Lot83].

This is of course false for p-tuples of words. $X = < a, c.c >$ and $Y = < a.a, c >$ give a counterexample.

2.1.2 Labelled transition systems

Labelled transition systems [Kel72, KM82, Arn90] yield a general formal framework which is often used to model the behaviour of concurrent processes. It is no more than a formalization of a simple structure which contains states and transitions linking them, those transitions being labelled with actions.

Formally, a labelled transition system $\mathcal{S}$ is a 4-tuple $< Q, A, T, S_0 >$ where

- Q is a set of *states*,

- A is a finite set of *actions* or *labels*,
- $T \subseteq Q \times A \times Q$ is a set of *transitions*,
- $S_0 \in Q$ is the *initial state*.

If a triple (S, a, S') is a transition of $\mathcal{S}$, we say that the action a is *fireable* in S and leads to S'. We shall write

$$S \xrightarrow{a} S'$$

Let S and S' in Q and $a_1, \ldots, a_n$ a sequence of actions in A, we will write

$$S \xrightarrow{a_1 \ldots a_n}{}^{*} S'$$

if there exists some states $S_1 = S, S_2, \ldots S_n, S_{n+1} = S'$ such that

$$\forall k \in [1:n], S_k \xrightarrow{a_k} S_{k+1}$$

A state $S' \in Q$ is said to be *reachable from* S if there exists $w \in A^*$ such that $S \xrightarrow{w}{}^{*} S'$. It is said to be *reachable* if it is reachable from S_0.

The whole behaviour of a transition system $\mathcal{S}$ can be described as a directed (possibly infinite) labelled tree, called the *reachability tree* and denoted by $RT(\mathcal{S})$. It is defined by the following rules:

- nodes are labelled with reachable states. We will use a small letter s for a node and the corresponding capital letter S for its labelling state.
- the root s_0 is labelled with the initial state S_0,
- edges are labelled with fireable actions.

The *reachability graph* $RG(\mathcal{S})$ is obtained from $RT(\mathcal{S})$ by merging nodes labelled with identical states. Thus its set of nodes is isomorphic to the set of reachable states.

For a given state $S \in GS$ we define the language $L(\mathcal{S}, S)$ which is the set of sequences of actions which are fireable from S:

$$L(\mathcal{S}, S) = \{w \in A^* \text{ s.t. } \exists S' \in GS, S \xrightarrow{w}{}^{*} S'\}$$

In particular $L(\mathcal{S}, S_0)$, the set of sequences of actions which are fireable from S_0 is called the *language of the system* and is denoted by $L(\mathcal{S})$.

2.1.3 Communicating finite state machines and Fifo-Nets

Communicating finite state machines

Systems of communicating finite state machines (Cfsm systems) [Boc78] are often used for the specification of communication protocols.

A Cfsm system consists of a set of finite state machines which communicate asynchronously by messages through fifo channels.

Formally, a Cfsm system is made up of a set of finite state machines $\{P_1, \ldots P_N\}$ associated with a set of fifo channels $f_{i,j}$, one for each pair of machines. To each channel $f_{i,j}$ is associated an alphabet $M_{i,j}$ of messages. These alphabets are supposed to be disjoint. Each machine P_i is a finite transition system $< Q_i, A_i, T_i, q_{0_i} >$ where:

- Q_i is a finite set of n_i states and q_{0_i} is the initial state.
- A_i is a finite set of actions. An action can either be an internal action (τ), or an output of a message m into a fifo $f_{i,j}$ ($-m$ for $m \in M_{i,j}$) or an input of a message m from a fifo $f_{j,i}$ ($+m$ for $m \in M_{j,i}$).
- $T_i \subseteq Q_i \times A_i \times Q_i$ is the set of *local transitions*.

The semantics that we consider is interleaving: if several actions of distinct machines are locally fireable, all possible interleavings must be considered as possible global behaviours. Moreover, we must also consider the non-deterministic choice between local actions. This semantics allows to consider the whole behaviour of the system as a labelled transition system $\mathcal{S} =< Q, A, T, S_0 >$ where:

- $Q = (Q_1 \times \cdots \times Q_N) \times (\times_{i,j=1..N, i \neq j} M^*_{i,j})$. Thus, a global state $S \in Q$ is composed of:
  - a N-tuple $E(S) =< E_1(S), \ldots E_N(S) >$ of local states of machines
  - a $(N^2 - N)$-tuple $C(S) =< C_{1,2}(S), \ldots C_{1,N}(S), \ldots C_{N,N-1}(S) >$ of fifo channels contents.
- $A = A_1 \cup \cdots \cup A_N$: a global action is an action of one of the machines,
- $S_0 =< q_{0_1}, q_{0_2}, \ldots, q_{0_N}, \epsilon, \ldots, \epsilon >$
- and $T \subseteq Q \times A \times Q$ is defined in the following way: let S and $S' \in Q$ be two global states and $a \in A$ be a global action. The triple t=(S,a,S') is in T if one the following conditions is satisfied:

  · $a = -m \in A_i$ for $m \in M_{i,j}$ (output of m from P_i to P_j through $f_{i,j}$) and

$$\begin{array}{l} (E_i(S), a, E_i(S')) \in T_i \text{ and } \forall k \neq i, E_k(S') = E_k(S) \\ C_{i,j}(S') = C_{i,j}(S).m \text{ and } \forall (k,l) \neq (i,j), C_{k,l}(S') = C_{k,l}(S) \end{array}$$

  · $a = +m \in A_j$ for $m \in M_{i,j}$ (input of m in P_j from $f_{i,j}$) and

$$\begin{array}{l} (E_j(S), a, E_j(S')) \in T_j \text{ and } \forall k \neq j, E_k(S') = E_k(S) \\ m.C_{i,j}(S') = C_{i,j}(S) \text{ and } \forall (k,l) \neq (i,j), C_{k,l}(S') = C_{k,l}(S) \end{array}$$

  · $a = \tau_i \in A_i$ (internal action in P_i) and

$$\begin{array}{l} (E_i(S), a, E_i(S')) \in T_i \text{ and } \forall k \neq i, E_k(S') = E_k(S) \\ C(S) = C(S') \end{array}$$

Figure 1 describes a system of two communicating finite states machines. A part of its reachability graph is shown in Figure 2. One can easily realize that its reachability graph is infinite. In fact, the initial pattern in grey is infinitely repeated with a strict increasing of channels contents.

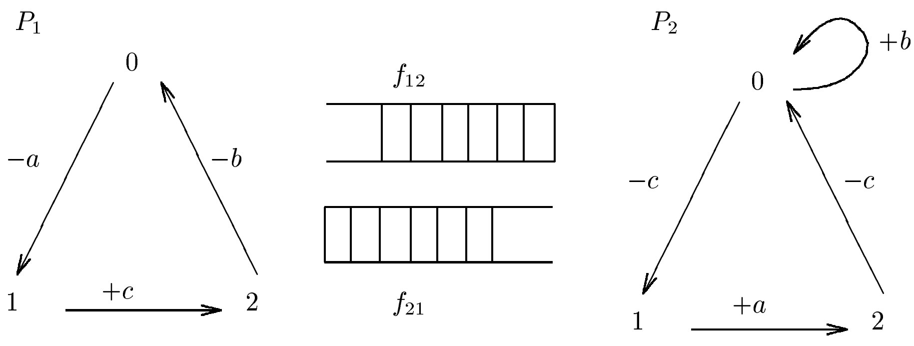


Figure 1: Example of Cfsm system

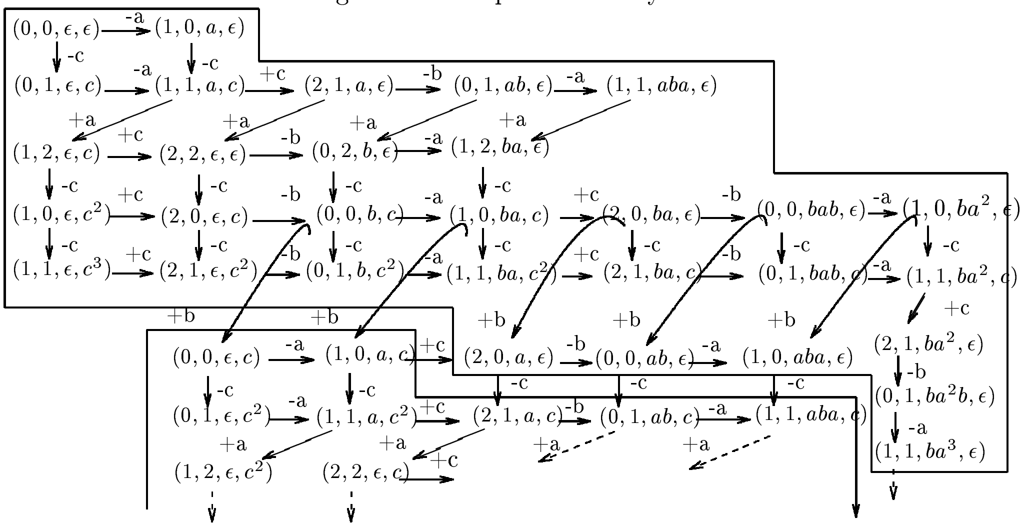


Figure 2: A part of the infinite reachability graph of the Cfsm system

Fifo-Nets

Fifo-Nets [MM81, FR88] are a generalization of Petri-Nets [Bra83]. Here, instead of places where tokens are added and removed, messages are concatenated and removed from fifo queues.

Formally, a Fifo-Net is a pair $< N, M_0 >$ where $N =< F, T, A, V >$ and

- F is a finite set $\{f_1, \ldots f_n\}$ of fifo queues,
- T is a finite set of transitions,
- A is a finite alphabet of messages,
- $V : F \times T \cup T \times F \to A^*$ is a valuation function
- M_0 is a function from F into A^*. Therefore, M_0 can be considered as a n-tuple of words. It is called the *initial marking.*

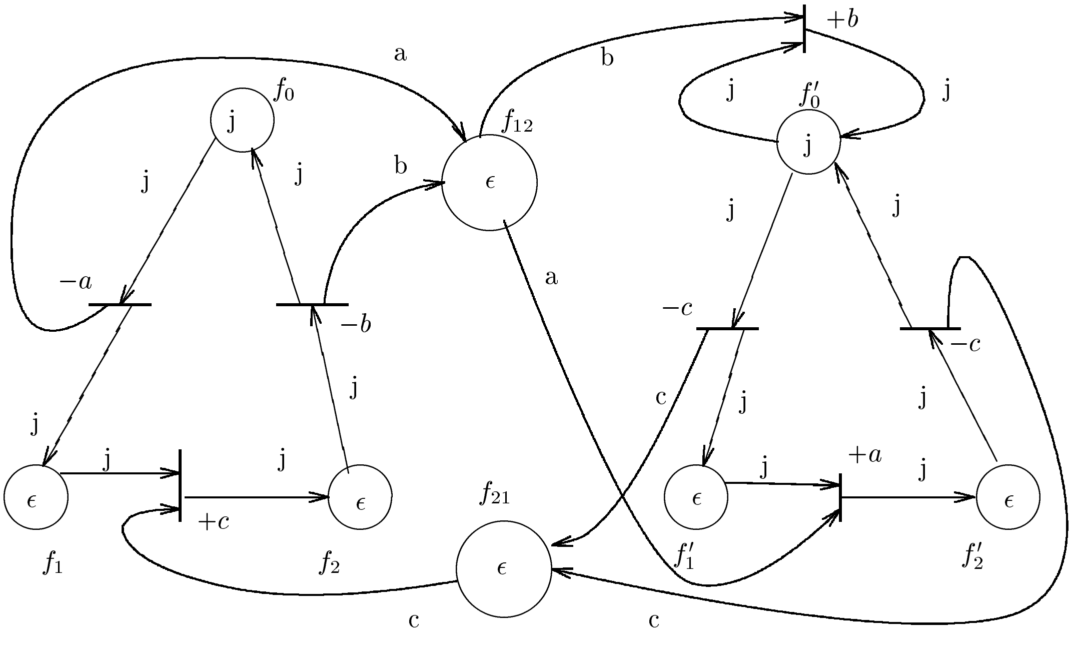


Figure 3: Example of Fifo-Net

A marking M is a function from F into A^* which represents the n-tuple of fifo queues contents $< M(f_1), \ldots, M(f_n) >$.

For such a marking M, we say that a transition $t \in T$ is *fireable* if and only if

$$\forall f \in F, V(f,t) \leq M(f) \text{ (for the prefix ordering in } A^*)$$

This is denoted by $M(t>$.

The *firing* of t in M then leads to a marking M' such that

$$\forall f \in F, \quad V(f,t).M'(f) = M(f).V(t,f)$$

We denote this $M(t > M'$.

The semantics of Fifo-Nets allows us to represent the whole behaviour of a Fifo-Net $< N, M_0 >$ in terms of a transition system.

Notice that Fifo-Nets are at least as powerful as Cfsm systems: every Cfsm system can be translated into a Fifo-Net with same behaviour. Figure 3 represents the Fifo-Net translated from the Cfsm system of Figure 1. It is represented as a bipartite graph, where circles and bars respectively represent fifo queues and transitions (here transitions are labelled with their corresponding actions in the Cfsm system). An edge from a queue f to transition t is labelled with $V(f,t)$ and an edge from a transition t to queue f is labelled with $V(t,f)$.

The initial marking M_0 is $M_0(f_0) = M_0(f'_0) = j$ and $M_0(f) = \epsilon$ for all other queues. Starting from M_0, the transition labelled by $-a$ is fireable and leads to the marking M such that $M(f_1) = j$, $M(f'_0) = j$, $M(f_{12}) = a$ and all other queues are empty.

2.1.4 Transitions systems with fifo channels

In the remainder of this paper we shall consider a particular form of transition systems general enough to describe the behaviours of Cfsm systems, Fifo-Nets and, with a few modifications, Estelle specifications. Since the main assumption that is made is the existence of fifo channels as the communication medium, we call them *transition systems with fifo channels* (TSFC).

A TSFC is a particular transition system $\mathcal{S} =< GS, A, T, S_0 >$ where

- GS is the set of states that we call *global states.*
  A global state S is a pair $< E(S), C(S) >$ where
  - $E : GS \to LS$, and LS is a finite set.
    $E(S)$ is called the *control state* of S.
  - $C : GS \to M_1^* \times \cdots \times M_M^*$ for finite alphabets of *messages* M_j associated to fifo channels f_j, $j \in [1:M]$.
    The M-tuple $C(S) =< C_1(S), \ldots C_M(S) >$ represents the channels contents.
- $S_0 =< E(S_0), C(S_0) > \in GS$ is the initial state.
- A is the finite set of actions.
  An action $a \in A$ is a triple $< \delta(a), in(a), out(a) >$ where
  - $\delta(a) : LS \to 2^{LS}$ is a function which denotes the modification of the control state during a firing of the action a.
  - $in(a) =< in_1(a), \ldots, in_M(a) >$ is a M-tuple of words denoting the inputs to be performed on channels while firing a.
  - $out(a) =< out_1(a), \ldots, out_M(a) >$ is a M-tuple of words denoting the outputs to be performed on channels while firing a.

in and out are considered as functions from A into $M_1^* \times \cdots \times M_M^*$. According to the definition, an action can contain several inputs and outputs of words upon the alphabets of messages.

- $T \subseteq GS \times A \times GS$ is the *transition relation.*
  A triple (S, a, S') is in T if the following conditions are satisfied:
  - a is *fireable*, that is
    - $\delta(a)(E(S)) \neq \emptyset$,
    - $in(a) \leq C(S)$ for the prefix ordering in $M_1^* \times \cdots \times M_M^*$ (all inputs are possible).
  - and the firing of a leads to S', which is satisfied if
    - $E(S') = \delta(a)(E(S))$ (denotes the modification of the control state),
    - $in(a).C(S') = C(S).out(a)$ (denotes the modification of channels contents according to the fifo mechanism).

  If (S, a, S') is in T, we will note $S \xrightarrow{a} S'$.

We define a partial ordering between global states which generalizes the prefix ordering between (p-tuples of) words:

Definition 1 *S is a prefix of S' denoted $S \leq S'$ if and only if the control states of S and S' are identical, and channels contents in S are prefix (between words) of those of S' i.e.*

$$S \leq S' \iff (E(S) = E(S') \text{ and } C(S) \leq C(S'))$$

We note $S < S'$ iff $S \leq S'$ and $S \neq S'$ (i.e. $\exists f_j$ such that $C_j(S) < C_j(S')$).

The two functions in and out are extended into functions on sequences of A^*. Each coordinate in_j (resp. out_j) is extended into a morphism of monoids from A^* into M_j^*.

The *input language* [Fin86] $L_I(f_j)$ of a fifo channel f_j is the set of message sequences filling up the channel f_j i.e.

$$L_I(f_j) = out_j(L(\mathcal{S}))$$

We will also use the family of languages $L_I(f_j, S) = out_j(L(\mathcal{S}, S))$ for $S \in GS$.

The input language can be used to define particular classes of transition systems:

Definition 2 *Let $L \subset A^*$ be a language. Denote by $\prod$ the concatenation product.*

$$L \text{ is monogeneous [Fin88] iff } L \subseteq \bigcup_{i=1}^{N} LF(u_i.v_i^*) \text{ with } u_i, v_i \in A^*$$

$$L \text{ is linear [CF87] iff } L \subseteq \bigcup_{i=1}^{N} \prod_{k=1}^{N_i} a_{i,k}^* \text{ where } a_{i,k}\text{'s are distinct letters of } A$$

$$L \text{ is word-linear iff } L \subseteq \bigcup_{i=1}^{N} LF(\prod_{k=1}^{N_i} w_{i,k}^*) \text{ where } w_{i,k}\text{'s are words of } A^*$$

A transition system is called monogeneous (resp. linear, word-linear) iff the input language of each fifo channel is monogeneous (resp. linear, word-linear).

Notice that linear and monogeneous languages are particular word-linear languages.

After a close look at the reachability graph of the Cfsm system of Figure 1 which reachability graph is represented in Figure 2, we can prove that this system is word-linear:

$L_I(f_{1,2}) \subseteq LF((ab)^*)$ is monogeneous and thus word-linear
$L_I(f_{2,1}) \subseteq c^*$ is linear, monogeneous and thus word-linear

Definition 3 *A fifo channel f_j is bounded iff there exists a constant integer K_j such that, for each reachable state S, the length of the content $C_j(S)$ of the fifo channel f_j is less than K_j.*

Kœnig's lemma asserts that every infinite tree of finite degree has an infinite branch. If $RG(\mathcal{S})$ is infinite then so is $RT(\mathcal{S})$. So, there exists at least an infinite sequence in $RT(\mathcal{S})$. If the nodes of each infinite sequence in $RT(\mathcal{S})$ are labelled with a finite number of different states, then $RG(\mathcal{S})$ is finite. As a consequence, $RG(\mathcal{S})$ is infinite if and only if there exists an infinite sequence of nodes of $RT(\mathcal{S})$ labelled with an infinite number of different states. But in the transition system defined above, control states can take at most a finite number of values and the only possibly unbounded objects are channels contents. So, we have:

Proposition 1 *The reachability graph $RG(\mathcal{S})$ of a TSFC is finite if and only if all channels are bounded [Pac82].*

2.2 Known decidability results about Cfsm and Fifo-Nets

Theorem 1 *The unboundedness problem is undecidable for two communicating finite state machines, Fifo-Nets and for transition systems with fifo channels.*

Sketch of proof: The result for Cfsm systems was proved by Brand and Zafiropulo [BZ81, BZ83]. It is a consequence of the capability for Cfsm systems to simulate Turing machines and of the undecidability of the halting problem.

In fact, it is a well known result that there exists a particular Turing machine Z_0, the universal machine, for which the halting problem is undecidable (there is no program that correctly answers the question "Does Z_0 stops with input tape X?" for every sequence X). Starting from that machine, one can build a system of two Cfsm $\mathcal{S}$ which behaviour simulates Z_0 and such that Z_0 stops if and only if every input action $a = -m$ of $\mathcal{S}$ is fireable (there exists a reachable global state from which a is fireable).

Now, if the unboundedness problem was decidable, one could decide if all input actions are fireable. But this contradicts the undecidability result of the halting problem for Z_0.

The inclusion of Cfsm systems into Fifo-Nets and into TSFC implies that the unboundedness problem is also undecidable for those models. But the result for Fifo-Nets was also proved by Finkel in [Fin82] with the use of language theory.

□

The first idea to bypass the undecidability of the unboundedness problem is to work with restrictive models. This gives some decidability results. For systems of two Cfsm, unboundedness is decidable in the following cases:

- one of the two fifo channels is bounded [BZ81, BZ83],
- one Cfsm is restricted to send only a single type of messages [GR84, RY86],
- the input language of one fifo is monogeneous [Fin86].

But, those results are too much restrictive because they cannot be generalized to systems of more than two Cfsm. However, there exists some decidability results for general Cfsm systems and Fifo-Nets:

- if all the queues alphabets consist of a single type of messages: it is a sub-class of Petri-Nets and then the decidability result is proved in [KM69].
- if all the input languages are linear [GGLR87, CF87]: such a system can be simulated with an extension of Petri nets called *Petri nets with structured sets of terminal markings* in which the unboundedness problem is still decidable. But a decision procedure can be exhibited without this translation. This procedure is based on an ordering $\ll$ between markings which counts the number and range of letters a_i of the input language in each fifo queue. This procedure works essentially because you can always decode the channel content over the letters a_i.
- if all the input languages are monogeneous [Fin86, Fin88]: in that class one can build a finite coverability graph of the (infinite) reachability graph which allows the decidability of the unboundedness problem, quasi-liveness and other reachability problems. It is based on the existence of an ordering $\ll$ between global states defined by

$$S \ll S' \iff \forall f_j, \forall y \in L_I(f_j, S),\ C_j(S).out_j(y) \le C_j(S').out_j(y)$$

2.3 Justifying the testing approach for unboundedness

We presented above the main decidability results for Cfsm systems and Fifo-Nets. But these results seem unsatisfactory for several reasons. As a matter of fact, the only not obvious ones for more than two Cfsm (and Fifo-Nets) are obtained for the classes of linear and monogeneous systems. But the membership to those classes is generally undecidable [Fin86]. Moreover, the decision procedures explicitly use the input languages. So they can only be used on systems for which monogeneous or linear languages including their input languages are known.

Suppose you are given a Cfsm system or Fifo-Net $\mathcal{S}$. You have to answer the undecidable question "*Is my system linear or monogeneous?*" and, if the answer is yes, "*what is the monogeneous or linear language including the input language of each fifo channel of $\mathcal{S}$?*" before using the decision procedure.

In the case of linear systems, the generalization of the decision procedure for word-linear systems seems impossible: even if the set of words $w_{i,k}$ is a code you generally cannot decode the channels contents into words.

To sum up, the above decision procedures seems too much restrictive because they need additional informations on channels which are generally undecidable. But, in counterpart, the monogeneous and linear classes of systems allow to decide some other problems in reachability analysis.

Our approach is conceptually different for we are looking for a sufficient condition for unboundedness, the weakest possible one, which could be applied to all specifications. The basis of this test is the construction of a *reduced tree* (some kind of Karp and Miller tree): begin as if you were constructing the (infinite) reachability tree, and stop a sequence if you find a node satisfying the condition or when a loop is detected.

A very important argument in favour of a testing approach in reachability analysis is that, because of the *state explosion problem*, decidability results, if they exist, are merely theoretical. The reachability graphs of real size systems, even if they are finite, are often very large. While generating those graphs, you must store an increasing number of states with a bounded memory. Thus, a decision procedure which uses this kind of construction gives one of the three results: the property is true, false or there is a memory overflow. And this is exactly what gives a test.

However, our search for a test is somewhat in the continuation of the decidability results for linear and monogeneous systems. Both were based on the construction of a reduced reachability tree built upon an ordering $\ll$ on global states having three essential properties [Fin87]:

- **monotonicity**: if $S \longrightarrow^* S'$ and $S \ll S', S \neq S'$, then for every action a, $S \xrightarrow{a} S_1 \Rightarrow S' \xrightarrow{a} S'_1, S_1 \ll S'_1, S_1 \neq S'_1$,
- **well-ordering**: in every infinite sequence $S_1 \ldots S_n \ldots$ of distinct states, there exists a strictly increasing sub-sequence, i.e. there exists indexes $i < j$ such that $S_i \ll S_j$,
- **computability** of the ordering and of the equality between states.

Monotonicity and computability gives a sufficient condition and well-ordering assures that the algorithm always stops.

3 The unboundedness test

3.1 Definition of a relation between global states

The generalized prefix ordering seems to be a good candidate in order to provide a sufficient condition for unboundedness. But we can easily see that it does not work: we can build finite graphs in which nodes s and s' satisfy $s \rightarrow^* s'$ and $S < S'$. See the example of Figure 4.

Therefore, we are looking for a minimum condition to add to this ordering in such a way that the resulting relation $\mathcal{U}$ is a sufficient condition for unboundedness.

We are then looking for such a relation $\mathcal{U}$ satisfying the following monotonicity property:

$$s \overset{w}{\rightarrow^*} s' \text{ and } \mathcal{U}(s, s'), S \neq S' \Longrightarrow \exists w' \in A^*, s' \overset{w'}{\rightarrow^*} s'' \text{ and } \mathcal{U}(s', s''), S' \neq S''$$

But the existence of w' is difficult to prove if we do not explicitly characterize it. However, in most cases, unboundedness is a consequence of the ability to infinitely repeat the same sequence of actions. So we will try to satisfy the stronger property:

$$s \overset{w}{\rightarrow^*} s' \text{ and } \mathcal{U}(s, s'), S \neq S' \Longrightarrow s' \overset{w}{\rightarrow^*} s'' \text{ and } \mathcal{U}(s', s''), S' \neq S'' \quad (1)$$

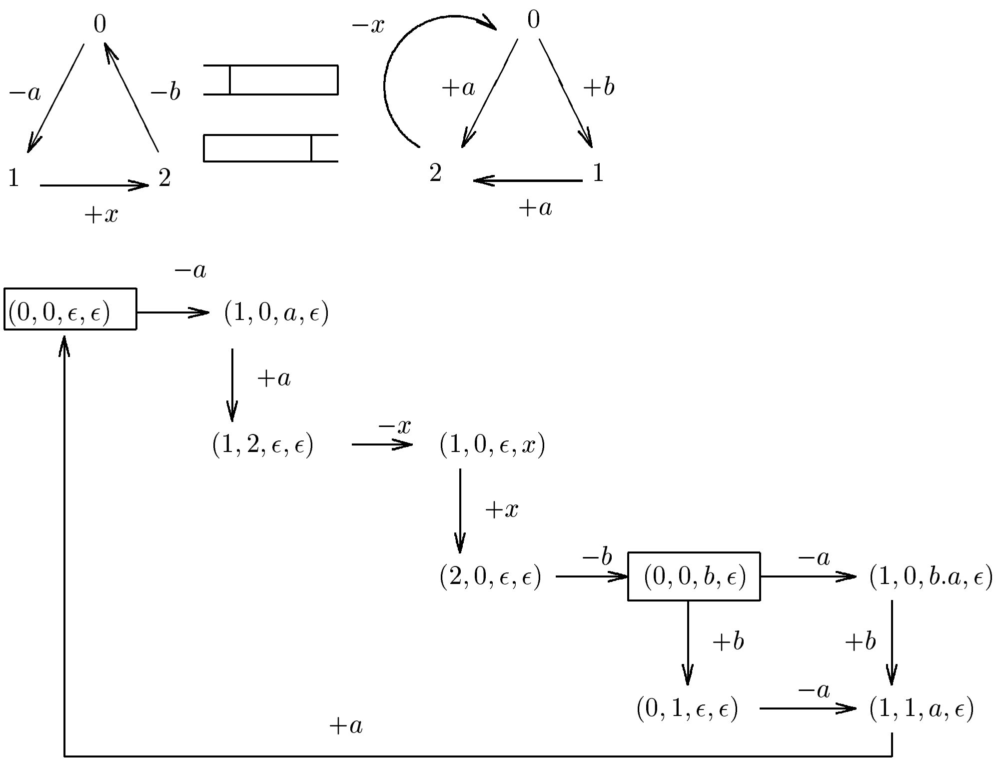


Figure 4: Cfsm system with a finite reachability graph in which two states are strictly ordered by the prefix ordering

Suppose that (1) is true and that the relation $\mathcal{U}$ is included in the prefix ordering $\leq$: $\mathcal{U}(s,s') \Rightarrow S \leq S'$. Then if

$$s \xrightarrow{w}{}^{*} s' \text{ and } \mathcal{U}(s,s')$$

we have

$$\exists Q \in M_1^* \times \ldots \times M_M^* \text{ s.t. } C(S') = C(S).Q$$

Now, $s' \xrightarrow{w}{}^{*} s''$ gives

$$\begin{aligned} in(w).C(S'') &= C(S').out(w) \\ &= C(S).Q.out(w) \end{aligned}$$

And $\mathcal{U}(s',s'')$ gives $C(S') \leq C(S'')$ which is equivalent to

$$in(w).C(S') \leq in(w).C(S'')$$

Thus
$$C(S).out(w) = in(w).C(S') \leq in(w).C(S'') = C(S).Q.out(w)$$
And finally, by removing $C(S)$
$$out(w) \leq Q.out(w)$$
As it is a consequence of (1), this inequality is a necessary condition for (1). Conversely we will prove in the next section that if we add this inequality to $\leq$ we have a sufficient condition for unboundedness. Thus this will prove that $out(w) \leq Q.out(w)$ is the minimum condition to add to $\leq$ in order to obtain a sufficient condition for unboundedness satisfying the property (1).

Lemma 1 *Let s and s' be two nodes of $RT(\mathcal{S})$ such that $s \overset{w}{\rightarrow}^* s'$ and $S \leq S'$. Then*
$$out(w) \leq Q.out(w) \text{ (where } Q \text{ is the } M\text{-tuple such that } C(S') = C(S).Q)$$
is equivalent to
$$C(S).out(w) \leq C(S').out(w)$$
As a consequence
$$S \leq S' \text{ and } out(w) \leq Q.out(w)$$
is equivalent to
$$E(S) = E(S') \text{ and } C(S).out(w) \leq C(S').out(w)$$

Proof: Suppose
$$out(w) \leq Q.out(w)$$
Then
$$C(S).out(w) \leq C(S).Q.out(w) = C(S').out(w)$$
Conversely suppose
$$C(S).out(w) \leq C(S').out(w)$$
This inequality implies that $C(S) \leq C(S')$ and thus there exists a M-tuple Q such that $C(S') = C(S).Q$. The inequality now gives
$$C(S).out(w) \leq C(S).Q.out(w)$$
and
$$out(w) \leq Q.out(w)$$

□

Let's now define the relation $\mathcal{U}$. It contains the well-ordering over monogeneous systems found in [Fin86]:

Definition 4 *If s and s' are two nodes of the reachability tree labelled with global states S and S' and such that $s \overset{w}{\rightarrow}^* s'$*
$$\mathcal{U}(s, s') \iff E(S) = E(S') \text{ and } C(S).out(w) \leq C(S').out(w)$$
We will note $\mathcal{U}_{\neq}(s, s')$ in the event of $\mathcal{U}(s, s')$ and $S \neq S'$.

As a consequence of the definition and with the same notations, we have:

Proposition 2 *If $s \xrightarrow{w} s'$ and $\mathcal{U}_{\neq}(s, s')$ then $out(w)$ and Q commute.*

Proof: Let $j \in [1..M]$ be a channel index.
We have

$$out_j(w) \leq Q_j.out_j(w)$$

Thus

$$\exists y \in {M_j}^* \text{ s.t. } Q_j.out_j(w) = out_j(w).y$$

On the other hand

$$C_j(S).out_j(w) = in_j(w).C_j(S') = in_j(w).C_j(S).Q$$

and $out_j(t)$ is shorter than Q_j. Thus

$$\exists z \in M_j \text{ s.t. } out_j(w) = z.Q_j$$

Now

$$Q_j.z.Q_j = Q_j.out_j(w) = out_j(w).y = z.Q_j.y$$

and $Q_j.z$ has the same length than $z.Q_j$, Thus $y = Q_j$ and finally

$$Q_j.out_j(w) = out_j(w).Q_j$$

This is true for each channel f_j, and then

$$Q.out(w) = out(w).Q$$

□

Remark: Q and $out(w)$ are not necessarily powers of the same M-tuple of words. But for each coordinate j, Q_j and $out_j(w)$ are powers of the same word.

3.2 The relation is a sufficient condition for unboundedness

Now, we prove that the relation $\mathcal{U}$ gives a sufficient relation for unboundedness.

We begin with the proof of a property which shows that the prefix ordering is not far from being a sufficient condition for unboundedness.

Proposition 3 *Let s and s' be two nodes in $RT(\mathcal{S})$ s.t. $S \leq S'$, and $a \in A$ a single action which is fireable in S and leads to S_1. Then a is fireable from S' and there exists a state S'_1 such that $S' \xrightarrow{a} S'_1$ and the control states of the two reached nodes S_1 and S'_1 are identical i.e.*

$$(s \xrightarrow{a} s_1 \text{ and } S \leq S') \Rightarrow (\exists s'_1 \text{ s.t. } s' \xrightarrow{a} s'_1 \text{ and } E(S_1) = E(S'_1))$$

Proof: From the definition of $s \xrightarrow{a} s_1$ and $S \leq S'$ we have

$$\begin{array}{l} E(S_1) \in \delta(a)(E(S)) \text{ and } in(a) \leq C(S) \\ in(a).C(S_1) = C(S).out(a) \\ E(S) = E(S') \text{ and } C(S) \leq C(S') \end{array}$$

Therefore
$$in(a) \leq C(S) \leq C(S') \text{ and } E(S_1) \in \delta(a)(E(S'))$$
Then a is fireable in S' and leads to all states S'_1 such that

$$\begin{array}{l} E(S'_1) \in \delta(a)(E(S')) = \delta(a)(E(S)) \\ in(a).C(S_1) = C(S_1).out(a) \end{array}$$

In particular, since $E(S_1) \in \delta(a)(E(S'))$, there exists a state S'_1 such that $E(S'_1) = E(S_1)$.

□

Remark: Proposition 3 is true for one single action a but cannot be generalized to a sequence of actions. In fact, we generally do not have $S_1 \leq S'_1$ and thus we cannot apply the same construction again.

We have some kind of monotonicity property:

Proposition 4 *Let $w = a_1 \dots a_n$ and $s, s' \in RT(\mathcal{S})$ such that $s \overset{w}{\rightarrow}{}^{*} s'$ and $\mathcal{U}_{\neq}(s, s')$. Denote by s_1 the node s.t. $s \overset{a_1}{\rightarrow} s_1$. Then the action a_1 is fireable in s' and leads to a node s'_1 such that $s' \overset{a_1}{\rightarrow} s'_1$ and $\mathcal{U}_{\neq}(s_1, s'_1)$.*

Proof: $\mathcal{U}(s, s')$ implies $S \leq S'$. Thus, Proposition 3 asserts that a_1 is fireable from s' and there exists a node s'_1 such that

$$s' \overset{a_1}{\rightarrow} s'_1 \text{ and } E(S'_1) = E(S_1)$$

Now if we note $w' = a_2 \dots a_n.a_1$, we have $s_1 \overset{w'}{\rightarrow}{}^{*} s'_1$.

Proposition 3 gives $E(S_1) = E(S'_1)$. So we only need to prove

$$C(S_1).out(w') < C(S'_1).out(w')$$

We have

$$in(a_1).C(S_1).out(w') = C(S).out(a_1).out(w') = C(S).out(w).out(a_1)$$

But if we denote by Q the M-tuple of words such that $C(S') = C(S).Q$,

$$out(w) < Q.out(w) \text{ implies } out(a_1) < Q.out(a_1)$$

Thus

$$\begin{array}{rcl} in(a_1).C(S_1).out(w') & = & C(S).out(w).out(a_1) \\ & < & C(S).out(w).Q.out(a_1) \\ & < & C(S).Q.out(w).out(a_1) \\ & < & C(S').out(w).out(a_1) \\ & < & C(S').out(a_1).out(a_2 \dots a_n).out(a_1) \end{array}$$

And finally
$$C(S_1).out(w') < C(S'_1).out(w')$$

$\square$

Theorem 2 *Let s and s' be two nodes of the reachability tree $RT(\mathcal{S})$ of a TSFC. If $s \xrightarrow{w}^* s'$ and $\mathcal{U}_{\neq}(s,s')$ then the reachability graph $RG(\mathcal{S})$ is infinite.*

Proof: Let $w = a_1 \ldots a_n$. Proposition 4 can be applied with s, s' and a_1 and again with s_1 and s'_1. If you reiterate this n times you build two sequences

$$s, s_1, \ldots, s_n = s' \text{ and } s', s'_1, \ldots, s'_n = s''$$

such that

$$s_i \xrightarrow{a_{i+1}\ldots a_n . a_1 \ldots a_i}{}^* s'_i \text{ and } \mathcal{U}_{\neq}(s_i, s'_i)$$

Thus

$$s' \xrightarrow{w}^* s'' \text{ and } \mathcal{U}_{\neq}(s', s'')$$

which implies $S' < S''$ for global states. You can now reiterate the same process and build an infinite sequence of nodes $s, s', s'', \ldots, s^{(k)}, \ldots$ such that the infinite sequence of labelling global states is strictly increasing for the prefix ordering.

Definition 5 *The (possibly infinite) reduced tree $RedT(\mathcal{S})$ is built like the reachability tree $RT(\mathcal{S})$ except for the following reduction: if $s' \in RT(\mathcal{S})$ and there exists a node s in the sequence from s_0 to s' such that $\mathcal{U}(s,s')$ then s' is marked and the subtrees starting from s' are not explored.*

3.3 Unboundedness decidability for word-linear systems

Theorem 3 *The reduced tree $RedT(\mathcal{S})$ of a word-linear TSFC $\mathcal{S}$ is finite. Thus the unboundedness problem is decidable for word-linear systems.*

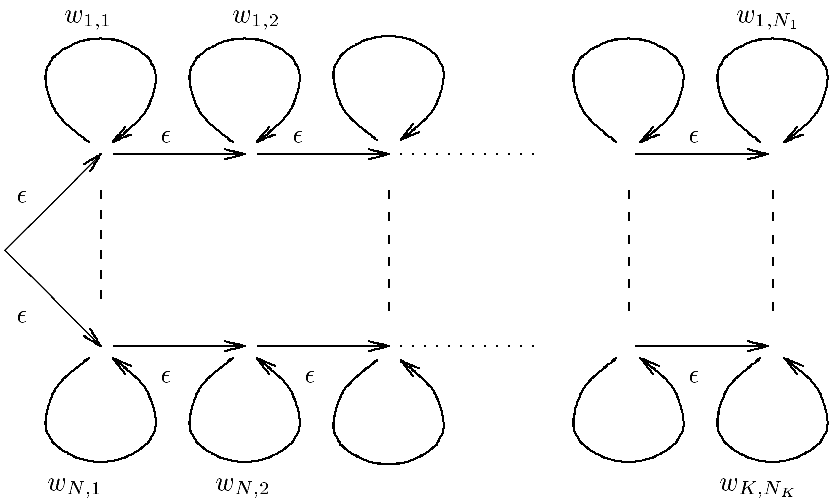
Proof: According to the Kœnig's lemma, we only need to prove that in every infinite transition sequence of $RT(\mathcal{S})$, there exist two nodes s and s' such that $s \rightarrow^* s'$ and $\mathcal{U}(s,s')$.

Consider an infinite sequence of nodes in $RT(\mathcal{S})$. The set LS of control states is finite. So we can extract from this sequence a finite number of sub-sequences such that the control states $E(S)$ in all these nodes are identical. Among those sub-sequences, at least one is infinite. Let $(s_m)_m$ be one of them.

For a particular channel f_j we have

$$L_I(f_j) \subseteq \bigcup_{i=1}^{N} LF(\prod_{k=1}^{N_i} w^*_{i,k})$$

An automaton of $L_I(f_j)$ is then included in the following one (all states are accepting ones):



Denote by v_m the sequence of actions from s_m to s_{m+1}. For each m, $out_j(v_m)$ can only take one of the "trajectories" defined by a particular index i in the above automaton. Now, by considering all possible behaviours of $out_j(v_1 \dots v_m)$ and $in_j(v_1 \dots v_m)$ it is easy to describe all possible contents of f_j in $(s_m)_m$. So, there exists at least one index $i \in [1:N]$ such that:

$$\forall m, C_j(S_m) \in \bigcup_{k=1}^{N_i} [LF(RF(C_j(S_0).w_{i,k})) \cup \bigcup_{l=k}^{N_i} (RF(C_j(S_0).w_{i,k}). \prod_{h=k}^{l} w_{i,h}^{*}.LF(w_{i,l}))]$$

For every pair (k,l) of indexes such that $1 \leq k \leq l \leq N_i$, we can find a sub-sequence $(s_n)_n$ of $(s_m)_m$ such that there exist

- sequences $(r_{k,n})_n \dots (r_{l,n})_n$ of integers
- two constants words $c \in LF(RF(C_j(S_0).w_{i,k}))$ and $d \in LF(w_{i,l})$

satisfying:

$$\forall n, C_j(S_n) = c.(\prod_{h=k}^{l} w_{i,h}^{r_{h,n}}).d$$

c and d are fixed parts and k, l and $(r_{k,n})_n \dots (r_{l,n})_n$ define the variable part $\prod_{h=k}^{l} w_{i,h}^{r_{h,n}}$ of $(s_n)_n$.

Remark that if there exist indexes n_1 and h_1 s.t. $r_{h_1,n_1} \neq 0$ then $\forall h < h_1$ the sequences $(r_{h,n})_{n>n_1}$ are stationary. In fact, $r_{h_1,n_1} \neq 0$ means that we have started to output the word w_{i,h_i}. Thus the preceding ones $w_{i,h}, h < h_1$ can't be output anymore.

Thus either from an index n_1, all sequences $(r_{h,n})_n$ are stationary and $(C_j(S_n))_{n>n_1}$ too, or there exists an index h_1 such that from $(r_{h_1,n})_n$ we can extract a strictly increasing sub-sequence $(r_{h_1,p})_p$. In the second case, according to the above remark, all sequences $(r_{h,p})_{h<h_1}$ are stationary and $(r_{h,p})_{h>h_1}$ are null (otherwise $(r_{h_1,p})_p$ would be stationary). Then $(C_j(S_p))_p$ is strictly increasing for the prefix ordering and every element of the sequence can be wrote

$$C_j(S_p) = C.w_{i,h_1}^{r_{h_1,p}}.D$$

with C a constant word in $LF(RF(C_j(S_0).w_{i,k})).\prod_{h=k}^{h_1-1} w_{i,h}^{r_h}$, and D a constant word in $LF(w_{i,h_1})$ and constant integers $r_k, \ldots, r_{h-1}$.

Denote by $Q_{j,p}$ the word s.t. $C_j(S_{p+1}) = C_j(S_p).Q_{j,p}$ and v_p the sequence of actions from s_p to s_{p+1}.

- if $k \neq h_1$ then $\forall p, in_j(v_p) = \epsilon$ and $out_j(v_p) = Q_{j,p}$.
  Thus
  $$C_j(S_p).out_j(v_p) < C_j(S_{p+1}).out_j(v_p)$$
- otherwise $C \in LF(RF(C_j(S_0).w_{i,h_1}))$. If there exists a finite number of indexes p such that $in_j(v_p) \neq \epsilon$ then there exists an index p_1 such that $\forall p > p_1, in_j(v_p) = \epsilon$ and we fall in the preceding case. Otherwise there is an index p_1 s.t.
  $$C_j(S_0) \leq in_j(v_0 \ldots v_{p_1})$$
  i.e. $C_j(S_0)$ is completely received.

  We can then take C in $RF(w_{i,h_1})$. Now denote by B and E the words s.t. $B.C = D.E = w_{i,h_1}$. Thus $out_j(v_p)$ has the form $(E.D)^{l_p}$.

  We have:
  $$C_j(S_p) = C.(B.C)^{r_{h_1,p}}.D = C.(D.E)^{r_{h_1,p}}.D$$
  and
  $$C_j(S_{p+1}) = C.(B.C)^{r_{h_1,p+1}}.D = C.(B.C)^{r_{h_1,p}}.(D.E)^{r_{h_1,p+1}-r_{h_1,p}}.D$$
  Thus, as $out_j(v_p) = (E.D)^{l_p}$ we conclude
  $$C_j(S_p).out_j(v_p) < C_j(S_{p+1}).out_j(v_p)$$

Then for the channel f_j we have found a sub-sequence $(s_p)_p$ such that the sequence $(C_j(S_p))_p)$ is either stationary, or strictly increasing for $\leq$ and satisfying

$$\forall p, C_j(S_p).out_j(v_p) < C_j(S_{p+1}).out_j(v_p)$$

We then apply the same construction with the resulting sub-sequence, consecutively with the other channels. Finally, this diagonalization builds an infinite sub-sequence in which two consecutive nodes satisfy $\mathcal{U}$.

4 Application

4.1 Implementation of the unboundedness test

Figure 5: Algorithm constructing reduced graphs

```
St_States:=nil;          (* stack of states of the current sequence *)
St_Fireable:=nil;        (* stack of sets of fireable actions *)
Visited:=nil;            (* heap of completely visited states *)
Marked:=∅;               (* set of marked states *)
push(S0, St_States);
push(fireable(S0),St_Fireable);
while St_States ≠ nil do begin
    cur_state:=top(St_States);
    if top(St_Fireable) ≠ ∅ then begin
        a:=take_one_elt_off(top(St_Fireable));
        successors:=succ(cur_state,a);   (* set of states reached by firing a *)
        forall S' ∈ successors do
            if S' ∉ St_States ∪ Visited then
                if ∃S ∈ St_States s.t. U(S,S') then(* UNBOUNDED *)
                    Marked:=Marked ∪{S'}
                else begin
                    push(S', St_States);
                    push(fireable(S'),St_Fireable);
                    end;
    else begin           (* top(St_Fireable) = ∅ *)
        pop(St_States);
        pop(St_Fireable);
        Visited:=Visited ∪{cur_state};
        end;
    end;
```

A prototype of the program of Figure 5 which constructs the reduced trees of extended Cfsm systems has been written in Pascal.

While testing unboundedness we only need to know the current sequence of actions and states from the root. We can then theoretically use a depth first strategy in which we only keep trace of this sequence in a stack. In order to avoid traversing again nodes with same labelling global states we use the rest of the memory for recording completely visited nodes. The reachability graph may be infinite, so, when the memory overflows we randomly replace nodes [Hol87, JJ89, JJ91].

The number of tests performed for each new node s' is growing with its depth in $RT(\mathcal{S})$. In order to avoid this and as we are only testing for unboundedness, we can bound the number of nodes s for which we test $\mathcal{U}(s,s')$. The increase of time needed for the test for each node s' is then bounded by a function independent on its depth.

Although it heavily depends on the graph structure, we can give the theoretical time needed for all the computations of $\mathcal{U}(s, s')$ for a given node s' as a function of its depth $d(s')$ in $RT(\mathcal{S})$. The number of nodes s in question is less than $d(s')$. Their depth range over $[0..d(s')-1]$. For each node s, checking the equality $E(S) = E(S')$ requires less than N comparisons.

Let $s \xrightarrow{w}^{*} s'$ and note

$$|C(S)| = Max_{j=1}^{M}|C_j(S)| \text{ and } |out(w)| = Max_{j=1}^{M}|out_j(w)|$$

Checking whether

$$C(S).out(w) \leq C(S').out(w)$$

needs less than $M.(|C(S')| + |out(w)|)$ message comparisons. For a given node s', the total number of comparisons $F(s')$ satisfies

$$F(s') \leq \Sigma_{d(s)=0}^{d(s')-1} M.(|C(S')| + |out(w)|)$$

We can reasonably suppose that

$$|C(S')| \leq K_1.d(s') \text{ and } |out(w)| \leq K_2.(d(s') - d(s))$$

Thus

$$F(s') \leq M.K_1.d(s')^2 + M.K_2.\frac{d(s')^2 + d(s')}{2}$$

The complexity of checking whether there exists an s such that $\mathcal{U}(s, s')$ for a given s' is then in $\mathcal{O}(d(s')^2)$. But the experimental results of the next section will be more instructive.

4.2 Examples

This section gives some examples of unbounded Cfsm systems. They were all tested with a prototype of the algorithm presented above, running on a SUN4 workstation.

The first example is given by the Cfsm system $\mathcal{S}_1$ of Figure 1. We saw that its reachability graph is infinite (see Figure 2). It is a word-linear system:

$$L_I(f_{1,2}) \subseteq LF((ab)^*) \text{ is monogeneous}$$
$$L_I(f_{2,1}) \subseteq c^* \text{ is monogeneous and linear}$$

Then its reduced tree in Figure 6 is finite (Theorem 3). It has only 34 nodes, 5 of which are marked.

The Cfsm system $\mathcal{S}_2$ of Figure 7 is also word-linear:

$$L_I(f_{1,2}) \subseteq LF((ab)^*.a.(ba)^*) \text{ is word-linear,}$$
$$L_I(f_{2,1}) \subseteq LF(cd)^* \text{ is monogeneous.}$$

Unboundedness is then decidable. The program generates the 1221 nodes of the reduced tree in less than 7s. It detects 247 marked states and spent 27% of the time in the computation of $\mathcal{U}$.

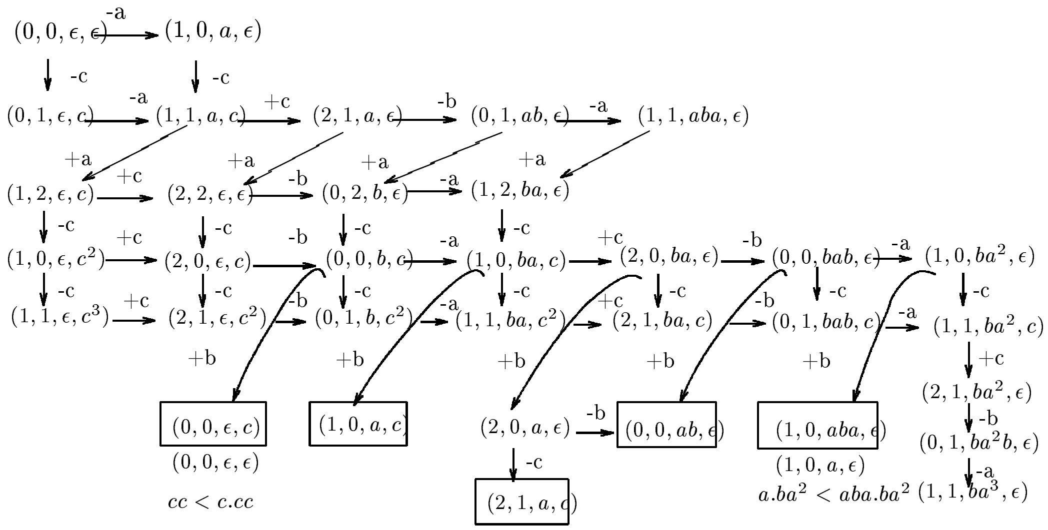


Figure 6: Reduced graph of the Cfsm system of figure 1

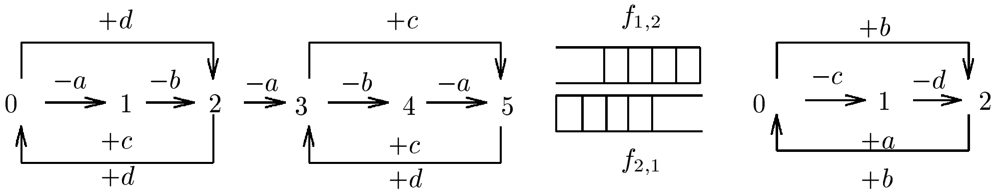


Figure 7: A word-linear system

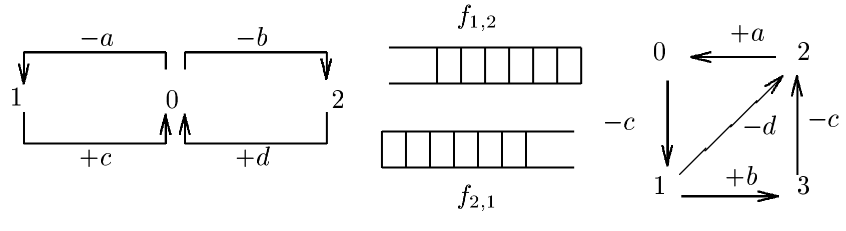


Figure 8: An unbounded system not detected as unbounded

The Cfsm system $\mathcal{S}_3$ of Figure 8, which produces an infinite reachability tree, has been specially designed to fail our test. There exists, among others, an infinite sequence of fireable actions of the form

$$w.v.u^2.v^2.\ldots.u^{2^n}.v^{2^n}\ldots$$

traversing an infinite number of increasing states (for the generalized prefix ordering) but the relation $\mathcal{U}$ is always false. This kind of "exponential growing" cannot be detected with our procedure. With channels bounded to 8 messages, the 1473 nodes of the finite graph are generated in 11 s. The time spent computing $\mathcal{U}$ is 50% of the total time.

Consider now a system obtained from $\mathcal{S}_3$ described in Figure 8 by identifying the two states 1 and 2 of the first Cfsm. Its reachability tree contains the reachability tree of $\mathcal{S}_3$. Thus there are infinite sequences in which the test is always false. But there are some other sequences in which the test is satisfied. We are in the case where unboundedness should be detected although the reduced tree is infinite.

4.3 Unboundedness of Estelle specifications

The Estelle language is a Formal Description Technique (FDT) standardized by ISO and well suited for the specification of distributed systems [90789]. A specification describes a hierarchy of modules which are essentially automata extended with Pascal constructs and which communicate by messages through fifo channels. Each local transition is supposed to be atomic thus, no assumption can be made concerning the relative speeds of modules. The semantics of the complete Estelle specification is defined in an operational way in terms of transformations of a *global state* (union of all local states, hierarchy of processes and channels contents). Several modes of parallelism can be described: sequential non-deterministic, synchronous or fully asynchronous. So, in order to model the behaviour of the specification, all possible interleavings of transitions must be considered as possible computations.

Unfortunately, we cannot directly apply the unboundedness test to Estelle specifications. In fact, the behaviour of an Estelle specification cannot be described by a TSFC. The first reason is that the set of possible global states is theoretically infinite, but this is not a real problem since we only test unboundedness. The main reason is that, due to existence of *priority* clauses, the possibility to fire a transition depends not only on the state of the module but also on other possible transitions. But we can avoid this problem too by adapting the notions of states and prefix ordering.

In order to apply our unboundedness test we must insure that the transition system of an Estelle specification satisfies the monotonicity property of Proposition 4. Recall that this property asserts that the relation $\mathcal{U}$ is preserved by the firing of a single action.

This proposition for TSFC is a consequence of Proposition 3:

$$(s \xrightarrow{a} s_1 \text{ and } S \leq S') \Rightarrow (\exists s_1' \text{ s.t.} s' \xrightarrow{a} s_1' \text{ and } E(S_1) = E(S_1'))$$

and the behaviour of channels contents.

The behaviour of channels contents in Estelle specifications is the same as in TSFC. Thus we only need to prove Proposition 3. The only problem we have to deal with is the existence of *priority* clauses used in transitions. This problem obliges us to redefine what is a global state and what is the prefix ordering between those states in such a way that

$$S \leq S' \Rightarrow \mathit{fireable}(S) \subseteq \mathit{fireable}(S')$$

where $\mathit{fireable}(S)$ is the set of fireable actions in S. As a matter of fact if no *priority* clause is used, global states are defined as in the ISO document (hierarchy of processes, variables, delayed values, ... and channels contents) and the prefix ordering is defined as the prefix ordering between channel contents and equality between the remainder. Otherwise, the set $\mathit{fireable}(S)$ must be included in the global state and the prefix ordering is intersected with the inclusion of the $\mathit{fireable}$ sets.

The definition of $\mathcal{U}$ and the proof of Proposition 4 are then almost identical.

5 Conclusion and prospects

Dealing with the difficult problem of unboundedness of fifo channels, we have defined a general transition system explicitly describing fifo channels and including Cfsm systems and Fifo-Nets.

We have exhibited a new relation between global states included in the prefix ordering and proved that it provides a sufficient condition for unboundedness. That relation induces a test for unboundedness by the construction of a reduced tree. The testing approach allows the detection of unboundedness even when the reduced tree is infinite. The test could also be improved if it was preceded by a static analysis of the system [BZ81] which could give bounds to some channels.

We proved that this test is, in a sense, strictly more powerful than the decision procedures for monogeneous and linear systems: the reduced tree of a word-linear system is always finite. Thus it provides a decision procedure for the class of word-linear systems. But quasi-liveness is still an open problem in this class. As a matter of fact, Our relation does not allow to build a coverability tree.

We think that the unboundedness test can be very useful for protocols designers in the verification phase of protocol specifications. It can easily be mixed with an on-line model-checking algorithm [JJ89] and then test unboundedness while, for example, verifying temporal logic formulas.

The applicability of our unboundedness test to Estelle specifications is, in our opinion, a very important result. As a matter of fact, Estelle is a real language for the specification of protocols and is already used in verification tools. Therefore, it would be interesting to provide the unboundedness test in such tools.

Acknowledgments The authors would like to especially thank Alain Finkel for all his remarks and the careful reading of the preliminary version of the paper. We also wish to thank the anonymous referees for their comments and suggestions.

References

[90789] ISO 9074. *Estelle: a Formal Description Technique based on an Extented State Transition Model.* ISO TC97/SC21/WG6.1, 1989.

[Arn90] A. Arnold. Systèmes de transitions finis et sémantique des processus communicants. *TSI*, 9(3):193–216, 1990.

[Boc78] G.V. Bochmann. Finite state description of communication protocols. *Computer Networks*, 2, October 1978.

[Bra83] G.W. Brams. *Réseaux de Petri: théorie et pratique.* Masson, 1983. Tome 1, 184 p.

[BZ81] D. Brand and P Zafiropulo. On communicating finite–state machines. Tech Rep. RZ 1053, IBM Zurich Research Lab., Ruschlikon, Switzerland, Jan. 1981.

[BZ83] D. Brand and P Zafiropulo. On communicating finite–state machines. *J.A.C.M*, 2:323–342, April 1983.

[CF87] A. Choquet and A. Finkel. Simulation of linear fifo nets by petri nets having a structured set of terminal markings. In *Proceedings of the* 8^{th} *European Workshop on Applications and Theory of Petri Nets, Zaragoza, Spain*, June 1987.

[Fin82] A. Finkel. Deux classes de réseaux à files: les réseaux monogènes et les réseaux préfixes. Thèse de $3^{\text{ème}}$ cycle, Rapport LITP, Oct. 1982.

[Fin86] A. Finkel. Structuration des systèmes de transitions. Applications au contrôle de parallélisme par files fifo. Thèse d'état, Juin 1986.

[Fin87] A. Finkel. A generalization of the procedure of Karp and Miller to well structured transition systems. In 14^{th} *ICALP , Karlsruhe, RFA*, July 1987.

[Fin88] A. Finkel. A new class of analysable cfsms with unbounded fifo channels: application to communication protocol and distributed solution of the mutual exclusion problem. In *VIII IFIP Symposium, WG 61, Atlantic City*, June 1988.

[FR88] A. Finkel and L. Rosier. A survey on decidability questions for classes of fifo nets. Rapport de recherche 456, L.R.I., Nov. 1988.

[GGLR87] M. Gouda, E. Gurari, T. Lai, and L. Rosier. On deadlock detection in systems of communicating finite state machines. *Computers and Artificial Intelligence*, 6(3):209–228, 1987.

[GR84] M. Gouda and L. Rosier. On deciding progress for a class of communicating protocols. In *Proceedings of the Eighteenth Annual Conference on Information Sciences and Systems*, pages 663–667, 1984.

[Hol87] G.J. Holzmann. Automated protocol validation in ARGOS, assertion proving and scatter searching. *IEEE trans. on Software Engineering, Vol 13, No 6*, June 1987.

[Je91a] T. Jéron. Contribution à la validation des protocoles : test d'infinitude et vérification à la volée. Ph.D Thesis, University of Rennes I, May 1991.

[Je91b] T. Jéron. Testing for unboundedness of fifo channels. In *STACS 91 : Symposium on Theoretical Aspects of Computer Science Hamburg, Germany*, february 1991. Springer–Verlag, LNCS #480, pages 322–333.

[JJ89] C. Jard and T. Jéron. On-line model-checking for finite linear temporal logic specifications. In *Proceedings of the International Workshop on Automatic Verification Methods for Finite State Systems*, Grenoble, France, June 1989. Springer–Verlag, LNCS #407, pages 275–285.

[JJ91] C. Jard and T. Jéron. Bounded memory algorithm for verification on-the-fly, Juin 1991. Publication interne Irisa n^o 588.

[Kel72] R.M. Keller. Vector replacement systems: a formalism for modeling asynchronous systems. Tech. Rep 117, Princeton Univ., 1972.

[KM69] R.M Karp and R.E Miller. Parallel program schemata. *Journal of Comput. System Sci.*, 3(2):147–195, 1969.

[KM82] T. Kasai and R.E. Miller. Homomorphisms between models of parallel computation. *J.C.S.S*, 25:285–331, 1982.

[Lot83] M. Lothaire. *Combinatorics on Words*, volume 17. Gian-Carlo Rota, Encyclopedia of Mathematics and its Applications, 1983.

[MM81] R. Martin and G. Memmi. Spécification et validation de systèmes temps réel à l'aide de réseaux de petri à files. Technical Report 3, Revue Tech. Thomson–CSF, Sept. 1981.

[Pac82] J. Pachl. Reachability problems for cfsms. Research Report CS–82–12, Univ. of Waterloo, Dept. of Comp. Sci., May 1982.

[RY86] L. Rosier and H. Yen. Boundedness, empty channel detection, and synchronization for communicating finite automata. In *T.C.S. 44*, pages 69–105, 1986.