Some Properties of Reactive Fiffo Automata

Frédéric Herbreteau

IRCCyN, France. Frederic.Herbreteau@irccyn.ec-nantes.fr

SUPERVISORS: Olivier Roux and Franck Cassez

KEYWORDS: Real-time systems, Reactive Fiffo Automata, Event memorizing, Boundedness.

Abstract. We are interested in the verification of real-time systems modelled with Reactive Fiffo[1] Automata. This model provides with the ability of memorizing events when they must not be taken into account at their occurrence date. We aim at deciding the boundedness of the queue.

1 Research Area – Main Themes

Real-time systems are often used for the control of critical systems like power plants, satellites, planes, etc. The critical aspects of all these systems enjoin to make sure that they meet some specifications, particularly safety requirements. This verification is usually achieved by model-checking, or at least testing the system.

We are interested here by real-time systems described with the ELECTRE formalism [4] : a language for the specification of reactive systems with asynchronous assumptions. Particularly, it provides with the ability of memorizing events in a queue when they must not be taken into account when they occur. This feature sets the problem of the boundedness of the queue. In this paper, we study this problem through *Reactive Fiffo*[1] *Automata* [4, 6, 2] (see Definition 1), the model for the execution of ELECTRE programs. Our goal is the detection of faulty systems having an unbounded queue.

1.1 Reactive Fiffo Automata

Reactive Fiffo Automata are built from a finite transition system (named the *control structure*) upgraded with a memorizing queue. They are very adequate for real-time systems modelling, since the fiffo memorizing policy extends the usual fifo in the following way : the event taken into account from the queue is the oldest of all the possible ones (it does not need to be at the beggining of the queue), see point (4c) in the following definition. Furthermore, the queue has always the priority among other transitions (expressed in points (4a), (4b) and (4c)).

[1] First In First Fireable Out.

Definition 1 (Reactive Fiffo Automaton). *A* Reactive Fiffo Automaton *(RFA) is a complete and deterministic transition system $R = (Q \times \Sigma_M^*, (q_0, \varepsilon), \Sigma \cup \{!, ?\} \times \Sigma_M, \delta)$, where :*

1. *Q is a finite set of locations,*
2. *$\Sigma = \Sigma_F \cup \Sigma_M$ is the alphabet of the RFA, divided in 2 subsets : the fleeting events (Σ_F), and the memorized events (Σ_M), such that $\Sigma_F \cap \Sigma_M = \emptyset$,*
3. *(q_0, ε) is the initial state,*
4. *$(q, w) \xrightarrow{\alpha} (q, w') \in \delta$ iff :*
   (a) *either $\alpha = e$ and $w = w'$ and $w \in (\Sigma_q^!)^*$,*
   (b) *or $\alpha = !e$ and $q = q'$ and $w' = we$ and $w \in (\Sigma_q^!)^*$,*
   (c) *or $\alpha = ?e$ and $\exists w_1, w_2 \in \Sigma_M^*$ s.t. $w = w_1 e w_2$ and $w' = w_1 w_2$ and $w_1 \in (\Sigma_q^!)^*$,*
5. *$\forall (q, w) \in Q \times \Sigma_M^*$, $\forall e \in \Sigma_M$:*
   (a) *either $(q, w) \xrightarrow{!e} (q, we)$,*
   (b) *or $\exists q' \in Q$ and $w' \in \Sigma_M^*$ s.t. $(q, w) \xrightarrow{e} (q', w)$ and $(q, w) \xrightarrow{?e} (q', w')$ w.r.t. (4c),*
6. *$\forall (q, w) \in Q \times \Sigma_M^*$, $\forall e \in \Sigma_F$: $\exists q' \in Q$ s.t. $(q, w) \xrightarrow{e} (q', w)$.*

with $\Sigma_q^! = \{e \mid q \xrightarrow{!e} q\}$.

One can find in [6] a sight seeing study of interesting properties of RFA (e.g., decidability of reachability). This model has also been used for the modelling and verification of an embedded software dedicated to the control program of an aircraft engine.

1.2 The Readers/Writers Example

As a running example, we consider the classical readers/writers problem. The corresponding RFA is depicted in Figure 1.

We consider two readers (R_1 and R_2) and a writer (W). The readers can access the book simultaneously, whereas the readers and the writer are concurrent. We denote $[\![R]\!]$ the set of all the executions of R; here is an example of execution :

$$(q_0, \varepsilon) \xrightarrow{w} (q_3, \varepsilon) \xrightarrow{!r_1} (q_3, r_1) \xrightarrow{!w} (q_3, r_1 w) \xrightarrow{r_2} (q_3, r_1 w r_2) \xrightarrow{end_W} (q_0, r_1 w r_2) \ldots$$
$$\ldots \xrightarrow{?r_1} (q_1, w r_2) \xrightarrow{?r_2} (q_4, w) \xrightarrow{end_{R_1}} (q_6, w) \xrightarrow{end_{R_2}} (q_0, w) \xrightarrow{?w} (q_3, \varepsilon) \ldots$$

Notice that the transition $(q_1, w r_2) \xrightarrow{?r_2} (q_4, w)$ clearly shows that the stored occurrences of events are processed as soon as possible (Fiffo ordering).

2 Directions of the Work

A RFA on its own describes an infinite number of executions : each time it memorizes an event, it can do it again without any limit. Thus, the boundedness problem is meaningful only when the environment of the RFA is constrained.

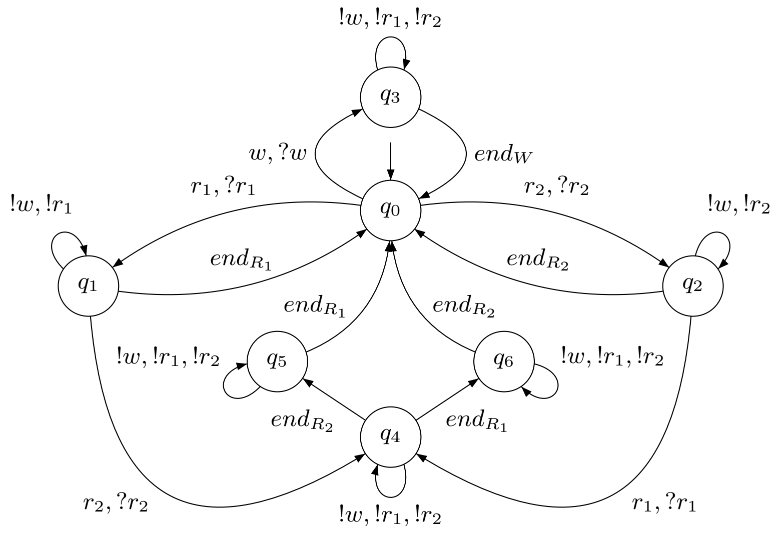


Fig. 1. Control structure of the RFA for the reader/writers.

2.1 Environment

In this paper, we consider that an environment is simply a regular language of infinite words :

Definition 2 (Environment). *An* environment $\mathcal{E}$ *is any ω-regular language :* $\mathcal{E} = \bigcup_i \{U_i V_i^\omega \mid U_i\ and V_i\ are\ regular\ languages\}$.

Then, the execution of a RFA R, in an environment $\mathcal{E}$ is defined by :

Definition 3 (Execution of a RFA in an environment). $[\![R, \mathcal{E}]\!] = \{\rho \in [\![R]\!] \mid \pi(\rho) \in \mathcal{E}\}$ *where π is the morphism on words defined by :* $\pi(e) = e$, $\pi(!e) = e$ *and* $\pi(?e) = \varepsilon$.

The definition of π clearly shows that the environment cannot control the processing of memorized events. Furthermore, $[\![R, \mathcal{E}]\!]$ always exists since the RFA is complete, and it is single since a RFA is deterministic with priority to the queue (full-deterministic).

2.2 Boundedness Problem

The boundedness problem is relevant for any system having a memorizing capability. It aims at deciding if a given system is bounded or not. For example, it has already been stated and solved for CFSM [1] and Petri Nets [5]. The following definition states this problem for RFA :

Definition 4 (Boundedness). *A Reactive Fiffo Automaton* $R = (Q \times \Sigma_M^*, (q_0, \varepsilon), \Sigma \cup \{!, ?\} \times \Sigma_M, \delta)$ *is* bounded *iff* $\exists k \in \mathbb{N}$ *s.t.* $\forall (q, w) \in Reach(R)$, $|w| \leq k$, *where $|w|$ denotes the length of word w, and $Reach(R)$ is the set of reachable states from (q_0, ε), that is :* $Reach(R) = \{(q, w) \in Q \times \Sigma_M^* \mid \exists \sigma \in (\Sigma \cup \{!, ?\} \times \Sigma_M)^*\ s.t.\ (q_0, \varepsilon) \xrightarrow{\sigma} (q, w) \in [\![R]\!]\}$.

For example, the RFA depicted on Figure 1, is bounded in the environment $\mathcal{E}_1 = \{w.end_W.r_2.end_{R_2}.w.end_W(r_1.w.end_{R_1}.end_W.r_2.end_{R_2}.w.end_W)^\omega\}$, then $k = 1$. But it is unbounded in the environment :
$\mathcal{E}_2 = \{w.r_2.w.end_W.r_1(end_{R_2}.w.end_{R_1}.r_2.w.r_1.end_W.w.r_2.w.end_W.r_1)^\omega\}$.

3 Results

In this section, we first study the boundedness problem in the context of ω-regular environments. Then, section 3.2 is dedicated to a particular class of regular environments : ultimately periodic words, which is relevant in the context of real-time systems. The following results are detailled in [3].

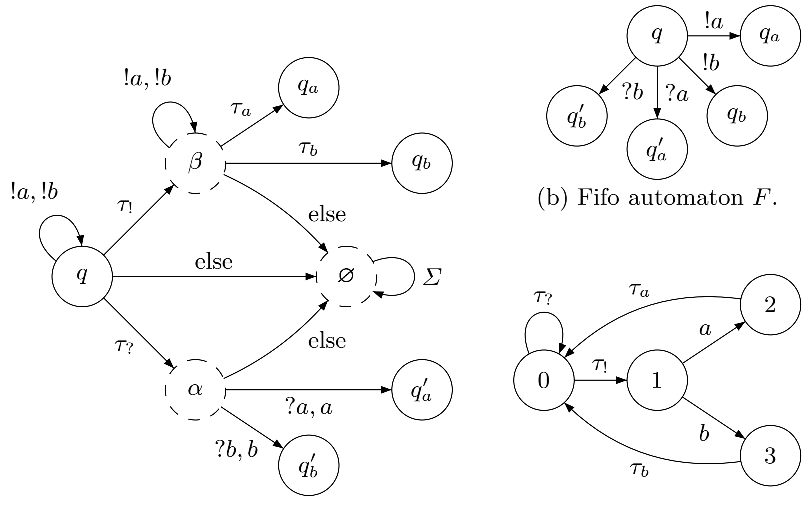
3.1 ω-Regular Environment

Since it is possible to simulate any fifo automaton [7] by a RFA coupled with an ω-regular environment, we have the following statement :

Theorem 1. *The boundedness problem is undecidable for RFA with ω-regular environments.*

Proof. Available from the author.

The construction of the RFA R and the ω-regular environment $\mathcal{E}$, simulating the fifo automaton F, is described in Figure 2. This algorithm should be repeated for each state of the fifo automaton F (namely q_a, q_b, q'_a and q'_b).



(a) Reactive fiffo automaton R.

(b) Fifo automaton F.

(c) ω-regular environment $\mathcal{E}$.

Fig. 2. Fifo automaton and the RFA and ω-regular environment simulating it.

Here are some details on our construction :

1. Each *else* transition in R stands for all the transitions that are not specified. Thus, an unspecified fleeting event τ (resp. memorized event e) leads to a transition labelled τ (resp. $?e, e$) towards state $\varnothing$ which is a deadlock state for the executions of R that are not executions of F.
2. The priority to the queue in R is bypassed by the introduction of two fleeting events, $\tau_?$ and $\tau_!$, thus allowing to choose between memorizing or taking into account from state q in R. In the same way, $\mathcal{E}$ can emit either $\tau_?$ or $\tau_!$ from its initial state.
3. Since RFA cannot change its state while memorizing, $\tau_!$ is also used to bring R in its "memorizing state" (namely β), while $\mathcal{E}$ evolves to state 1. Then, it emits either a or b which are memorized in β by R (thus allowing the same choice than F in q). Finally, the fleeting events τ_a or τ_b differentiate the memorizing of a or b respectively.
4. We must respect the fifo policy : in state α, a or b must be taken into account iff it begins the queue (despite the fiffo policy). This is achieved by the *else* transition (see point (1) above).
5. Finally, we picture states α, β and $\varnothing$ with dashed circles in order to highlight the fact that they have no corresponding states in F. Notice also that state β can be merged in state q, but this would make the construction harder to understand.

3.2 Ultimately Periodic Environment

The boundedness problem remains open for this particular class of ω-regular environments which consists of a single infinite word of the form : $w_0(w_p)^\omega$ where w_0 and w_p are finite words.

Since the environment is ultimately periodic (u.p.), one can answer if it is also the case for the execution of the RFA. We have proved that if the RFA is bounded, or if $Card(\Sigma_M) \leq 1$, then its execution is always u.p. However, it is not always the case; indeed, the RFA $R_\neg$ depicted in Figure 3, together with the environment $\mathcal{E}_\neg = (e_2\tau e_3\tau)^\omega$ has an aperiodic execution ($\odot$ denotes the usual concatenation on words) :

$$[\![R_\neg, \mathcal{E}_\neg]\!] = (!e_2\tau!e_3\tau) \cdot \bigodot_{n=1}^{\infty}\big((?e_2!e_2\tau!e_3\tau)^n \cdot (?e_3(?e_3)^n!e_2\tau!e_3\tau)\big)$$

The decision of the ultimate periodicity of the execution of a RFA in an u.p. environment is still open. We are also working on necessary and sufficient conditions for the infinite iteration of a cycle occuring in the execution of a RFA. Since computing the evolution of the size of the queue on a given finite sequence of transitions is purely syntactic, solving these two problems would give us either an algorithm (decidability of the first problem) or a test (undecidability of the first problem) for the boundedness problem.

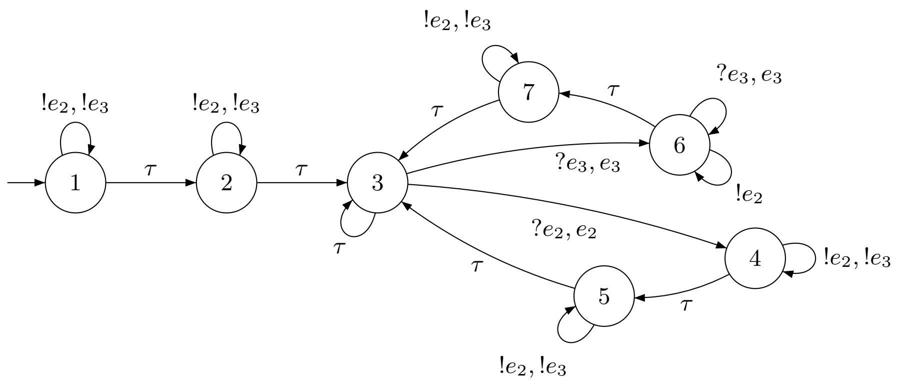


Fig. 3. Aperiodic reactive fiffo automaton.

4 Conclusion and Perspectives

The main encoutered difficulties rely on the fact that usual models (e.g., CFSM [1], Fifo automata [7]) are definable by morphisms, whereas RFA cannot because of the fiffo policy.

Future work will lead to the definition of the environment by temporal specifications (e.g., frequency of events occurrences), thus the study of timed/hybrid RFA. For example, the environment $\mathcal{E}_2$ (see Section 2.2) can be obtained with the following specifications : r_1 occurs 2 time units after r_2, r_2 occurs 2 t.u. after r_1, w occurs 1 t.u. after r_1 and r_2, and finally, activities R_1, R_2 and W last, respectively, 2, 1 and 3 t.u.

Aknowledgements We want to thank Grégoire Sutre and Alain Finkel (LSV - ENS Cachan, France) for their participation to this work, particularly for the idea of the proof of Theorem 1.

References

1. D. Brand and P. Zafiropulo. On communicating finite-state machines. *Journal of the ACM*, 30(2):323–342, April 1983.
2. F. Cassez, F. Herbreteau, and O. Roux. Reactive systems with unbounded event memorization. CARI'2000 Conference, Antananarivo, Madagascar, October 2000.
3. F. Cassez, F. Herbreteau, O. Roux, and G. Sutre. About boundedness of reactive fiffo automata. Submitted.
4. F. Cassez and O. Roux. Compilation of the ELECTRE reactive language into finite transition systems. *Theoretical Computer Science*, 146(1–2):109–143, July 1995.
5. J. L. Peterson. *Petri Net Theory and the Modeling of Systems.* Prentice Hall Int., 1981.
6. G. Sutre, A. Finkel, O. Roux, and F. Cassez. Effective recognizability and model checking of reactive fiffo automata. In *Proc. 7th Int. Conf. Algebraic Methodology and Software Technology (AMAST'98), Amazonia, Brazil, Jan. 1999*, volume 1548 of *Lecture Notes in Computer Science*, pages 106–123. Springer, 1999.
7. B. Vauquelin and P. Franchi-Zannettacci. Automates à file. *Theoretical Computer Science*, 11(2):221–225, June 1980. Note.