LMS J. Comput. Math. 14 (2011) 99–107 © 2011 Author
doi:10.1112/S1461157010000136

Muller condition and fairness on multitransition systems

Douadi Mihoubi

Abstract

This paper presents an extension of a result by Guessarian and Niar to the framework of multitransition systems. In the case of a single process, Guessarian and Niar had shown that the set of fair computations of regular SCCS processes coincides with the class of ε-free ω-regular languages. Here, in the case of multitransition systems, we show essentially that the sets of fair computations on multitransition systems are strictly included in the class of ε-free ω-regular N-languages. The inclusions of these fair sets into the class of ε-free ω-regular N-languages are obtained by showing that the strict (respectively weak, strong) fair condition can be simulated by the Muller acceptance condition on multitransition systems. The strictness of the inclusions is obtained by exhibiting two counter-examples showing that the reverse is false, that is, not every ω-regular N-language is the set of fair computations of some multitransition system.

1. *Introduction*

This paper is an extension of a result in Guessarian and Niar [4], proving that the set of fair computations of regular SCCS processes coincides with ε-free ω-regular languages. These results are obtained using so-called Muller T-automata, with infinitary transitions instead of infinitary states, which still recognize the class of ω-regular languages. This paper is focused to extend these results to the framework of multitransition systems defined by Arnold and Nivat. We show essentially the following results.

(1) The sets of fair computations of multitransition systems are included in the class of ε-free ω-regular N-languages, which extends the result for the case of a single process.

(2) Contrary to the case of a single process, fair computations do not capture all ε-free ω-regular N-languages. This result is obtained by exhibiting two counter-examples proving the strictness of the inclusions.

The paper is organized as follows. Basic notions and preliminaries on infinitary languages are presented in Section 2. After that, Section 3 is devoted to the definitions of fair conditions and the main result obtained by Guessarian and Niar on regular SCCS processes. The main results of this paper are given in Section 4. Finally, Section 5 is reserved for a concluding remark and an open question about the properties that a language from the class ε-free ω-regular N-languages must satisfy in order to be the set of fair computations of some multitransition system.

2. *Preliminaries*

Let Σ be a finite set of symbols which we call actions. A finite word over Σ is a finite sequence $w = (a_1, a_2, \ldots, a_n)$ of elements of Σ denoted by the concatenation $w = a_1a_2 \ldots a_n$. The integer $n = |w|$ is the length of the word w. The empty sequence ( ) of length 0 is called the empty word and is denoted by ε. The set Σ^* of all words over Σ equipped with the operation of concatenation defined by

$$a_1a_2 \ldots a_n \circ b_1b_2 \ldots b_m = a_1a_2 \ldots a_nb_1b_2 \ldots b_m$$

Received 15 April 2010; revised 4 November 2010.

2000 Mathematics Subject Classification 68Q10, 68Q85 (primary).

This work was supported by the LMPA, University of M'sila, Algeria.

https://doi.org/10.1112/S1461157010000136 Published online by Cambridge University Press

has the structure of a monoid with the empty word ε as a neutral element, called the free monoid on Σ. A language over Σ is any subset of Σ^*, which permits us naturally to use the Boolean operations over the set of languages $\wp(\Sigma^*)$. The concatenation product of Σ^* may be extended to $\wp(\Sigma^*)$ by defining $AB = \{ab/a \in A, b \in B\}$ for A, B in $\wp(\Sigma^*)$.

We write $\mathbb{N}^+ = \{1, 2, \ldots\}$. An infinite word over Σ is an infinite sequence $u : \mathbb{N}^+ \longrightarrow \Sigma$. We denote by $u[n]$ the finite word $u(1)u(2)\ldots u(n)$ representing the initial segment of u of length n. We denote by Σ^ω the set of infinite words over the alphabet Σ. We let $\Sigma^\infty = \Sigma^* \cup \Sigma^\omega$ be the set of finite or infinite words on Σ.

The concatenation product of Σ^* is extended to Σ^∞ by the following two rules.

(1) If $f \in \Sigma^*$ and $u \in \Sigma^\omega$, the product $fu \in \Sigma^\omega$ is defined by

$$fu(n) = \begin{cases} f(n) & \text{for all } 1 \leqslant n \leqslant |f|, \\ u(n - |f|) & \text{for all } n > |f|. \end{cases}$$

(2) If $u \in \Sigma^\omega$ and $\alpha \in \Sigma^\infty$, then $u\alpha = u$.

With this product, the set Σ^∞ has a structure of a monoid with the empty word ε as a neutral element. We call any subset L of Σ^* a *language*, any subset L of Σ^ω an *ω-language* and any subset of Σ^∞ an *infinitary language*.

A word $\beta \in \Sigma^\infty$ is a left factor of a word $\alpha \in \Sigma^\infty$, denoted by $\beta \leqslant \alpha$, if there exists $\gamma \in \Sigma^\infty$ such that $\alpha = \beta\gamma$. The relation $\leqslant$ is a partial order on Σ^∞ called the prefix ordering. We denote by $\mathrm{LF}(\alpha)$ the set of finite left factors of $\alpha \in \Sigma^\infty$.

Hence,

$$\mathrm{LF}(\alpha) = \{g/g \leqslant \alpha\} \quad \text{if } \alpha \in \Sigma^*$$

and

$$\mathrm{LF}(\alpha) = \{\alpha[n]/n \in \mathbb{N}^*\} \quad \text{if } \alpha \in \Sigma^\omega.$$

For any $L \subset \Sigma^\infty$, we let $\mathrm{LF}(L) = \{\mathrm{LF}(\alpha)/\alpha \in L\}$ denote the set of left factors of words in L.

The *adherence* of $L \subset \Sigma^\infty$ is defined by

$$\mathrm{Adh}(L) = \{\alpha \in \Sigma^\omega/\mathrm{LF}(\alpha) \subseteq \mathrm{LF}(L)\}$$

and L is said to be closed if $\mathrm{Adh}(L) \subseteq L$.

Given an increasing prefix ordering sequence $u_0 \leqslant u_1 \leqslant \ldots \leqslant u_n \leqslant \ldots$ of elements $u_i \in \Sigma^\infty$, there exists a unique element $\alpha = \lim_{n \longrightarrow \infty} u_n$ such that $k = |u_n|$ and $\alpha[k] = u_n$. Then $\alpha = u_n \in \Sigma^*$ if, for all $i \geqslant n$, we have $u_{i+1} = u_i$ or $\alpha \in \Sigma^\omega$. For any language $A \subset \Sigma^*$, we denote the *infinite iteration* of A by

$$A^\omega = \left\{ u \in \Sigma^\omega/u = \prod_{i \geqslant 1} u_i, \text{with } u_i \in A - \{\varepsilon\}, \text{for all } i \in \mathbb{N}^+ \right\}.$$

If $A = \{\varepsilon\}$, then $A^\omega = \{\varepsilon\}$.

We define the closure $\overrightarrow{A}$ of A by

$$\overrightarrow{A} = \lim A = \{u \in \Sigma^\omega/\text{for infinitely many } n, \text{we have } u[n] \in A\}.$$

In other words, $x \in \overrightarrow{A}$ if there is an increasing prefix ordering sequence $(a_n)_{n \in \mathbb{N}}$ of words in A such that $x = \lim_{n \longrightarrow \infty} a_n$.

A set $A \subset \Sigma^*$ is called prefix if, for all words x, y in A, if $x \leqslant y$, then $x = y$. In other words, the set A satisfies $A(\Sigma^* - \{\varepsilon\}) \cap A = \varnothing$.

EXAMPLE 1. For the set $X = ba^*$ over $\Sigma = \{a, b\}$, we have $X^\omega = (ba^*)^\omega$ and $\overrightarrow{X} = ba^\omega$. The set $Y = a^*b$ is prefix. On the contrary, the set $X = ba^*$ is not.

https://doi.org/10.1112/S1461157010000136 Published online by Cambridge University Press

3. *SCCS processes and fairness*

A process is any entity capable of performing computations. For more notions on models of computation, see [3]. When dealing with interacting systems (concurrent or distributed), the notion of fairness is an important concept which allows for example each process, waiting for a shared resource with other processes, to make progress in its computation. Three kinds of fairness have been studied in [4] by Guessarian and Niar on Milner's *SCCS* processes. We recall shortly Milner's calculus *SCCS* and main results obtained by these authors, showing that any ε-free ω-regular language is the language of (weakly, strongly or strictly) fair computation of some strict *SCCS* process, and vice versa.

Definition 1. An *SCCS* expression **E** is defined by the normal form scheme

$$E ::= x/\mathrm{Nil}/a : E/E \restriction B/E + F/E \times F/\mathrm{rec}\ \overrightarrow{x} \cdot \overrightarrow{F},$$

where $\mathrm{Act} = (A, \cdot, -, 1)$ is a non-trivial commutative group of actions with 1 the unit action of the group representing an internal action (the unit action 1 results from a synchronization $a\bar{a}$ or a delay of one time unit with $a \in A$), A is a finite set of actions and co-actions (actions and co-actions represent two complementary interactions between two processes), B is a subset of A, E, F are *SCCS* expressions and Nil is the inaction process. The operators $+$, $\times$ represent respectively the non-deterministic choice and synchronous product of two processes that perform an action simultaneously. The notation $E \restriction B$ is the restriction of process E to the actions of B and $a : E$ is the prefixing of E by the action a. Finally, x is a member of the set Var of variables and rec $\overrightarrow{x} \cdot \overrightarrow{F}$ denotes the solution of the equations $x_i = F_i$ for $i = 1, \ldots, n$ (the recursion operator is for us to be able to define processes with possibly infinite runs).

The algebra of *SCCS* processes is also equipped with the delay operator δ defined by $\delta E = \mathrm{rec}\ x \cdot (1 : x + E)$, which is the process that can either delay or behaves as E. Any process in *SCCS* is defined by only one equation. A component of the process E is any factor of a synchronous product $\times$ (for instance, the components of $E = p_1 \times (p_2 + p_3)$ are exactly p_1 and $(p_1 + p_2)$). A process is said to be active if and only if it performs an action different from the delay 1. A process E is said to be strict if all its subprocesses of the form $\mathrm{rec}\ x \cdot F$ contain neither occurrences of the synchronous product $\times$ nor unguarded variables within F. Recall that a synchronous product $p \times q$ cannot perform an action unless both p and q perform an action simultaneously.

Example 2. Let $\mathrm{Act} = (\{a, \bar{a}, b, \bar{b}, \}, \cdot, -, 1)$; then $q = \mathrm{rec}\ x \cdot \delta(a : x + b : x)$ is an *SCCS* process with behaviour $c(q) = (a + b)^{\omega} \cup \{\varepsilon\}$.

Definition 2. A computation of a process E is strictly fair (respectively strongly fair or weakly fair) if and only if every component of E (respectively any component which is infinitely often or continuously enabled from some point on) shall eventually be active.

Theorem 1 (Guessarian and Niar [4]). *Any ε-free infinitary language L is the language of (weakly, strongly or strictly) fair computations of some strict SCCS process, and vice versa.*

4. *Fairness and systems of N-processes*

In this section, we will formulate the definitions of fair computations described above in terms of a system of N-processes in order to study their behaviours according to these concepts as conditions of acceptance. The modelling of concurrent processes used here was introduced in [1] by Arnold and Nivat, which is based on regular infinitary languages. Each process is defined

https://doi.org/10.1112/S1461157010000136 Published online by Cambridge University Press

by its set of finite and infinite behaviours, and is modelled using an extended finite automaton to finite and infinite words. The parallelization of a collection of N-processes $(P_i)_{i=1,\ldots,N}$ is modelled by a concurrent automaton: the cartesian product of the automaton associated to each process P_i, called a multitransition system.

Definition 3. A system of N-processes is a fourtuple

$$S = (Q, \Sigma, \text{vec}\, q_0, T),$$

where:

1. $Q = Q_1 \times Q_2 \times \ldots \times Q_i \times \ldots \times Q_N$ is a finite set of states, with Q_i the set of states of the ith process;
2. $\text{vec}\, q_0 = (q_{01}, q_{02}, \ldots, q_{0i}, \ldots, q_{0N})$ is the start vector state of S with q_{0i} the start state of the ith process;
3. $\Sigma = (\Sigma_1 \cup \{\varepsilon\}) \times (\Sigma_2 \cup \{\varepsilon\}) \times \ldots \times (\Sigma_i \cup \{\varepsilon\}) \times \ldots \times (\Sigma_N \cup \{\varepsilon\})$ is a finite set of actions of S, with Σ_i the set of actions of the ith process;
4. $T \subseteq Q \times \Sigma \times Q$ is a finite set of transitions of S.

For each $i \in \{1, \ldots, N\}$, the set of states Q_i of the ith process is finite.

A transition of the system S is an element of $Q \times \Sigma \times Q$. The action performed during one transition will be an element $\overrightarrow{a}$ of $\overrightarrow{\Sigma}$ with

$$\overrightarrow{\Sigma} = (\Sigma_1 \cup \{\varepsilon\}) \times (\Sigma_2 \cup \{\varepsilon\}) \times \ldots \times (\Sigma_i \cup \{\varepsilon\}) \times \ldots \times (\Sigma_N \cup \{\varepsilon\}) - (\varepsilon)^N,$$

such that, for all $i = 1, \ldots, N$,

$$\Pi_i(\overrightarrow{a}) = \begin{cases} \varepsilon \text{ if the process } p_i \text{ is inactive during this interval time,} \\ \text{the action of } \Sigma_i \text{ executed by } p_i \text{ otherwise} \end{cases}$$

where Π_i denotes the projection on the ith component of the vector $\overrightarrow{a} \in \overrightarrow{\Sigma}$. The system S considered here is proper, that is, $(Q \times \overrightarrow{\varepsilon} \times Q) \cap T = \emptyset$, where $\overrightarrow{\varepsilon} = (\varepsilon, \varepsilon, \ldots, \varepsilon)$ is the N-empty word which represents the inaction of all N-processes in the same time.

Notation. To simplify, we denote by $\text{vec}\, q_j$ the vector state $(q_{j1}, q_{j2}, \ldots, q_{jN})$ and by $\overrightarrow{a_j}$ the vector action $(a_{j1}, a_{j2}, \ldots, a_{jN})$.

Let

$$p = \text{vec}\, q_0 \xrightarrow{\overrightarrow{a_1}} \text{vec}\, q_1 \xrightarrow{\overrightarrow{a_2}} \text{vec}\, q_2 \ldots \xrightarrow{\overrightarrow{a_i}} \text{vec}\, q_i \ldots$$

be an infinite path in the system S; then $\prod_{i=1}^{n} \overrightarrow{a_i}$ is a finite behaviour of p and $\prod_{i\geqslant 1} \overrightarrow{a_i}$ is the infinite behaviour of p. Let $t = (t_i)_{i\geqslant 0}$ be the sequence of consecutive transitions in the path p with $t_i = (\text{vec}\, q_i, \overrightarrow{a_i}, \text{vec}\, q_{i+1})$ and $r = (\text{vec}\, q_i)_{i\geqslant 0}$ its corresponding run; we denote by $\inf_t(p)$ the set of all transitions executed infinitely often in p and by $\inf_s(r)$ the set of all states entered infinitely many times in the run r.

Definition 4. A Muller multitransition system is a 6-tuple

$$S = (Q, \Sigma, \text{vec}\, q_0, T, Q_{\text{fin}}, Q_{\text{inf}}),$$

where:

1. $S_1 = (Q, \Sigma, q_0, T)$ is a system of N-processes;
2. $Q_{\text{fin}} \subseteq Q$ are the subsets of final states;
3. $Q_{\text{inf}} \subseteq 2^Q$ are the subsets of infinite states (designated subsets).

Multitransition systems were introduced to recognize languages and to realize processes [1]. The *behaviour* L of S will be a vector $(L_1, L_2 \ldots, L_N)$ of N regular infinitary languages,

https://doi.org/10.1112/S1461157010000136 Published online by Cambridge University Press

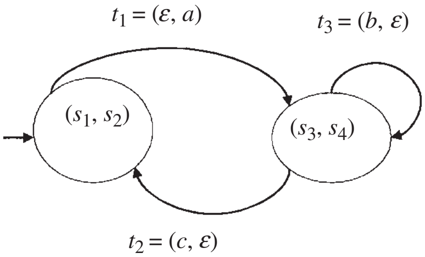

FIGURE 1.

where L_i is the behaviour of the ith process. The *behaviour* L_i of each process i will be a union $L_i^{\mathrm{fin}} \cup L_i^{\mathrm{inf}}$ of two regular languages, where L_i^{fin} is its regular finitary part and L_i^{inf} is its regular purely infinitary part.

We recall from [2] this useful theorem.

THEOREM 2. *Let A be a finite alphabet. The language $L \subseteq A^\infty$ is regular if and only if its finitary part L^{fin} is a regular language and its purely infinitary part L^{inf} is a finite union of the form*

$$L^{\mathrm{inf}} = \bigcup_{i=1}^{n} K_i R_i^\omega,$$

where, for all $i = 1, \ldots, n$, K_i, R_i are regular languages of finite words.

In [4], the equivalence between classical Muller automata and Muller T-automata is proved. Note that in Muller T-automata, the designated subsets are subsets of transitions instead of states. This result is also true for multitransition systems; henceforth we will consider multitransition systems with subsets of transitions as distinguished subsets.

4.1. *Strictly fair computations*

Let p be a path in a system of N-processes $S = (Q, \Sigma, \operatorname{vec} q_0, T)$; then p is called strictly fair if and only if:

- either p is a maximal finite path;
- or p is an infinite path which satisfies:

$$\forall i \in \{1, \ldots, N\}, \text{if there exists } t = (\operatorname{vec} q, \overrightarrow{a}, \operatorname{vec} q') \in \inf_t(p), \text{ then } \Pi_i(\overrightarrow{a}) \neq \varepsilon,$$

where $\inf_t(p)$ denotes the set of transitions entered infinitely many times in the path p.

We denote by

$$C_1(S) = \{w \in (\overrightarrow{\Sigma})^\infty / \exists \text{ a path } p \in \mathrm{pth}(\operatorname{vec} q_0, w) \text{ such that } p \text{ is strictly fair}\}$$

the set of strictly fair computations of S, with

$$\mathrm{pth}(\operatorname{vec} q_0, w) = \left\{ \begin{array}{c} t_{i_1} t_{i_2} \ldots t_{i_n} \ldots \in T^\infty / \forall i_j \in \mathbb{N}, t_{i_j} = (\operatorname{vec} q_{i_j}, \overrightarrow{a_j}, \operatorname{vec} q_{i_{j+1}}) \in T, \\ \text{where } \operatorname{vec} q_{i_1} = \operatorname{vec} q_0 \end{array} \right\}$$

the set of paths starting in $\operatorname{vec} q_0$ and having the behaviour

$$w = \overrightarrow{a_1}\overrightarrow{a_2} \ldots \overrightarrow{a_n} \ldots \in (\overrightarrow{\Sigma})^\infty.$$

EXAMPLE 3. Consider the above system of 2-processes $P_1 \times P_2$, where $\{s_1, s_3\}$ designate the state set of P_1 and $\{s_2, s_4\}$ that of P_2 (Figure 1).

https://doi.org/10.1112/S1461157010000136 Published online by Cambridge University Press

It is obvious that the path $c_1 = (t_1t_2)^\omega$ is strictly fair while $c_2 = t_1t_3^\omega$ is not: the process P_2 is delayed infinitely in c_2; however, in c_1 both processes P_i, $i = 1, 2$, perform alternately a different action from the delay ε.

PROPOSITION 1. *For every system of N-processes S, there exists a Muller multitransition system S_0 such that the infinitary N-language recognized by S_0 is equal to the set of strictly fair computations in S; in others, words $L^\infty(S_0) = C_1(S)$.*

Proof. Let $S = (Q, \Sigma, \text{vec}\, q_0, T)$ be a system of N-processes. A Muller multitransition system S_0 is constructed as follows:

$$S_0 = (Q, \Sigma, \text{vec}\, q_0, T, Q_{\text{fin}}, T_{\text{inf}}),$$

with:

(1) $Q_{\text{fin}} = \{\text{vec}\, q \in Q/\text{out}(\text{vec}\, q) = \emptyset \text{ in } S\}$ is the set of final states with $\text{out}(\text{vec}\, q) = \{\overrightarrow{a} \in \overrightarrow{\Sigma}/\exists\, \text{vec}\, q' \in Q : (\text{vec}\, q, \overrightarrow{a}, \text{vec}\, q') \in T\}$;

(2) $$T_{\text{inf}} = \left\{ \begin{array}{c} T' \subseteq T/T' = \{t_i = (\text{vec}\, q_i, \overrightarrow{a_i}, \text{vec}\, q_i')/i \in \{1, \ldots, l\}\} \in \text{cycle}(S) \text{ with} \\ \Pi_j\left(\bigcup_{i\in l} \overrightarrow{a_i}\right) \neq \varepsilon, \forall j \in \{1, \ldots, N\} \end{array} \right\}$$

are the subsets of infinite transitions, where:

(a) $\text{cycle}(S)$ is the set of cycles on the system of N-processes S;
(b) N is the number of processes;
(c) l is the number of transitions in the cycle T'.

In others words, the cycle T' is chosen to belongs to T_{inf} if at least one action different from the delay ε is executed by each process $j \in \{1, \ldots, N\}$ in the cycle T'.

We must show: u is a successful path in S_0 if and only if u is a strictly fair path in S.

We first show the direct case ($\Rightarrow$):

(a) if u is a successful finite path in S_0, this implies that u is of the form

$$u : q_0 \xrightarrow{\overrightarrow{a_1}} q_1 \xrightarrow{\overrightarrow{a_2}} q_2 \ldots \xrightarrow{\overrightarrow{a_n}} q_n$$

such that $q_n \in Q_{\text{fin}}$. Then $\text{out}(q_n) = \emptyset$ (by construction) and therefore u is a finite maximal path on S and hence u is strictly fair;

(b) if u is a successful infinite path in S_0, then there exists $T \in T_{\text{inf}}$ such that $T = \text{Inf}_t(u)$ with $T = \{t_i = (\text{vec}\, q_i, \overrightarrow{a_i}, \text{vec}\, q_i')/i \in \{1, \ldots, l\}\} \in \text{cycle}(S)$.

The set T forms a cycle of length l and we have $\Pi_j(\bigcup_{i\in l} \overrightarrow{a_i}) \neq \varepsilon$ (by construction). This shows that any process $j \in \{1, \ldots, N\}$ performs on T at least one action different from the delay ε; consequently, the path u is strictly fair.

For the converse ($\Leftarrow$):

if u is a strictly fair path on S, then two cases arise:

(1) the path u is finite and maximal on S and therefore u is successful in S_0 (by construction);
(2) u is a strictly fair infinite path on S, and hence the cycle T of transitions met infinitely often, in the path u, is a distinguished set, that is, $T \in T_{\text{inf}}$ and therefore u is successful in S_0.

□

https://doi.org/10.1112/S1461157010000136 Published online by Cambridge University Press

A direct consequence of this result is the following corollary.

Corollary 1. *The strictly fair language of a system of N-processes $S = (Q, \Sigma, \text{vec}\, q_0, T)$ is an infinitary regular N-language.*

Proof. The previous proposition shows that for any system of N-processes S, one can construct an equivalent Muller multitransition systems S_0. According to the known results on extended Muller automata and Muller T-automata, we have $C_1(S) = L(S_0) = L_0 \cup L_1$ with L_0 a regular N-language and L_1 an ω-regular N-language. □

4.2. *Weakly fair computations*

Let p be a path in a N-process $S = (Q, \Sigma, \text{vec}\, q_0, T)$; then p is called weakly fair if and only if:

- either p is a maximal finite path;
- or p is an infinite path which satisfies:

for all $i \in \{1, \dots, N\}$ if, for all $\text{vec}\, q \in \text{Inf}(c)$, we have $\Pi_i(\text{out}(\text{vec}\, q)) \neq \varepsilon$, then there exists $t_s = (\text{vec}\, q_s, \overrightarrow{a}, \text{vec}\, q'_s) \in \text{Inf}_t(u)$ such that $\Pi_i(\overrightarrow{a}) \neq \varepsilon$.

In other words, if the process P_i is continuously enabled from some point on the path, p shall eventually be active. We denote by $C_2(S)$ the set of weakly fair computations on S.

Example 4. In the system of 2-processes cited in Example 3 (Figure 1), the paths $c_1 = (t_1 t_2)^\omega$ and $c_2 = t_1 t_3^\omega$ are both weakly fair. In c_1 the processes (P_1, P_2) are continuously enabled and active, while in c_2 the only process which is continuously enabled is the process P_1.

Proposition 2. *For every system of N-processes S, there exists a Muller multitransition system S_0 such that the infinitary N-language recognized by S_0 is equal to the set of weakly fair computations of S; in other words, $L^\infty(S_0) = C_2(S)$.*

Proof. We use the same proof as in the case of strict fairness, Proposition 1, where T_{inf} is defined now by

$$T_{\text{inf}} = \left\{ \begin{array}{c} T' \subseteq T/T' = \{t_i = (\text{vec}\, q_i, \overrightarrow{a_i}, \text{vec}\, q'_i)/i \in \{1, \dots, l\}\} \in \text{cycle}(S) \\ \text{s.t. } \forall j \in \{1, \dots, N\}, \text{if } \bigcap\limits_{i \in \{1,\dots,l\}} \Pi_j(\text{out}(\text{vec}\, q_i) \neq \varepsilon) \Rightarrow \Pi_j \left(\bigcup\limits_{i \in l} \overrightarrow{a_i} \right) \neq \varepsilon \end{array} \right\}.$$

□

From this result, we can also deduce the following corollary.

Corollary 2. *The weakly fair language of a system of N-processes $S = (Q, \Sigma, \text{vec}\, q_0, T)$ is an infinitary regular N-language.*

4.3. *Strongly fair computations*

Let p be a path in the N-process $S = (Q, \Sigma, \text{vec}\, q_0, T)$; then p is called strongly fair if and only if:

- either p is a maximal finite path;
- or p is an infinite path which satisfies:

for all $i \in \{1, \dots, N\}$, if there exists $\text{vec}\, q \in \text{Inf}(c)$ with $\Pi_i(\text{out}(\text{vec}\, q)) \neq \varepsilon$, then there exists $t_s = (\text{vec}\, q_s, \overrightarrow{a}, \text{vec}\, q'_s) \in \text{Inf}_t(u)$ such that $\Pi_i(\overrightarrow{a}) \neq \varepsilon$.

In other words, if the process P_i is infinitely enabled from some point on the path, p shall eventually be active. We denote by $C_3(S)$ the set of strongly fair computations of S.

https://doi.org/10.1112/S1461157010000136 Published online by Cambridge University Press

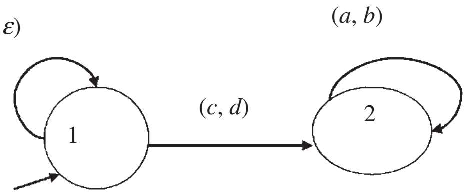

FIGURE 2.

PROPOSITION 3. *For every system of N-processes S, there exist a Muller multitransition system S_0 such that the infinitary N-language recognized by S_0 is equal to the set of strongly fair computations of S; in other words, $L^\infty(S_0) = C_3(S)$.*

Proof. We use the same construction as in the previous cases (the strict and weak fairness), but in this case the set T_{inf} is defined by

$$T_{\text{inf}} = \left\{ \begin{array}{c} T' \subseteq T/T' = \{t_i = (\text{vec}\, q_i, \overrightarrow{a_i}, \text{vec}\, q'_i)/i \in l\} \in \text{cycle}(S) \\ \text{s.t., } \forall j \in \{1, \dots, N\} \left(\bigcup_{i \in \{1,\dots,l\}} \Pi_j(\text{out}(\text{vec}\, q_i) \neq \varepsilon) \Rightarrow \Pi_j \left(\bigcup_{i \in \{1,\dots,l\}} \overrightarrow{a_i} \right) \neq \varepsilon \right) \end{array} \right\}. \qquad \square$$

From this result, we can also deduce directly the following corollary.

COROLLARY 3. *The strongly fair language of a system of N-processes $S = (Q, \Sigma, \text{vec}\, q_0, T)$ is an infinitary regular N-language.*

REMARK 1. Note that if a path in a system of N-processes S is strictly fair then it is also strongly and weakly fair.

4.4. *Muller acceptance condition and fairness*

In this section, we give two counter-examples showing that the converse of Proposition 10 (respectively Propositions 13 and 15) is not true, that is, not every ω-regular N-language is the set of fair computations of some system of N-processes.

4.4.1. *Strict fairness and Muller condition.* Let $M = P_1 \times P_2$ be the system of two processes (Figure 2), where all the subsets of the set of states $\{1, 2\}$ are considered to be designated.

The behaviour of this system is

$$L = L^\infty(M) = (a, \varepsilon)^* + (a, \varepsilon)^*(c, d)(a, b)^* + (a, \varepsilon)^\omega + (a, \varepsilon)^*(c, d)(a, b)^\omega.$$

It is obvious that $L \in \text{Rat}(\Sigma)^\infty$ is the class of infinitary regular languages. The question that arises is: does there exist a finite system S of 2-processes (P_1, P_2) such that $C_1(S) = L$? The infinite term $(a, \varepsilon)^\omega$ in L shows that the process P_2 is infinitely delayed. This shows that $(a, \varepsilon)^\omega$ can never be a strictly fair infinite behaviour of a certain finite system of 2-processes. Consequently, there is no S such that $C_1(S) = L$.

4.4.2. *Weak fairness and Muller condition.* Let M_1 be the multitransition system (Figure 3), where again all the subsets of $\{1, 2\}$ are considered to be designated. We have

$$L = L^\infty(M_1) = E_1 + E_2 + E_3 \quad \text{with}$$
$$E_1 = (a, \varepsilon)^*,$$
$$E_2 = (a, \varepsilon)^*(c, d)((a, b) + (\varepsilon, b))^*,$$
$$E_3 = (a, \varepsilon)^\omega + (a, \varepsilon)^*(c, d)((a, b) + (\varepsilon, b))^\omega.$$

https://doi.org/10.1112/S1461157010000136 Published online by Cambridge University Press

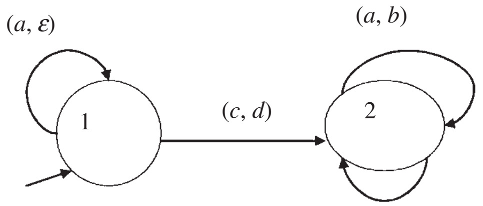

FIGURE 3.

The question that arises also in this case is: are there finite systems of 2-processes S such that $L = C_2(S)$? Suppose that such a system exists, and consider the infinite word $w = (a, \varepsilon)^*(c, d)(\varepsilon, b)^\omega \in L$.

From the hypothesis, we have $w \in C_2(S)$; consequently, there is in S a weak fair path σ of corresponding run c, such that $\Pi_2(\sigma) = w$. Let $\mathrm{vec}\, q \in \inf(c)$ be the state met in c after computing the prefix $(a, \varepsilon)^*(c, d)$ of w. But, we have $w_1 = (a, \varepsilon)^*(c, d)(\varepsilon, b)^*(a, b)^\omega \in C_2(S)$; this shows that $\Pi_1(\mathrm{out}(\mathrm{vec}\, q)) = \{a, \varepsilon\} \neq \varepsilon$; hence, the path σ is not weak fair, and therefore the system S in question cannot exist.

4.4.3. *Strong fairness and Muller condition.* Note that the same counter-example cited above for the weak fairness shows that there is no system of 2-processes S such that $L = C_3(S)$.

5. *Concluding remark*

We have shown that all kinds of fairness cited above are simulable by the Muller acceptance condition on multitransition systems but the reverse is false. According to Theorem 2 and the counter-examples cited in § 4.4, the behaviour $L \subseteq (\overrightarrow{\Sigma})^\infty$ of any system of N-processes, with any kind of fairness cited above as acceptance conditions, is an infinitary regular N-language; this means that the class $\mathrm{Fair}(\overrightarrow{\Sigma})^\infty$ of all fair sets on systems of N-processes is strictly included in the class $\mathrm{Rat}(\overrightarrow{\Sigma})^\infty$ of infinitary regular N-languages. This concluding result leads to the following theoretical question: what are the properties that a language from $\mathrm{Rat}(\overrightarrow{\Sigma})^\infty$ must satisfy in order to belong to the subclass $\mathrm{Fair}(\overrightarrow{\Sigma})^\infty$. Apparently, a closed infinitary regular language is not necessarily a set of fair computations of some system of N-processes.

Acknowledgements. The author would like to thank the editor and Professor I. Guessarian. He also wishes to offer many thanks to the anonymous referee for many corrections and suggestions which led to an improvement of the content of the paper.

References

1. A. Arnold and M. Nivat, 'Comportements de processus', *Colloque AFCET, les mathématiques de l'informatique*, Paris (1982), 35–68.

2. S. Eilenberg, *Automata languages and machines*, vol. A (Academic Press, New York, 1974).

3. M. Fernandez, *Models of computations, an introduction to computability theory* (Springer, Berlin, 2009).

4. I. Guessarian and W. Niar, 'Fairness and regularity for SCCS processes', *RAIRO Inform. Theor.* 23 (1989) no. 1, 59–86.

Douadi Mihoubi
Department of Mathematics, LMPA
M'sila University
P.O. Box 166, M'sila 28000
Algeria

mihoubi_douadi@yahoo.fr

https://doi.org/10.1112/S1461157010000136 Published online by Cambridge University Press