Distributed Computing manuscript No.
(will be inserted by the editor)

Randomized Dining Philosophers Without Fairness Assumption ⋆

Marie Duflot, Laurent Fribourg, Claudine Picaronny

LSV, CNRS & ENS de Cachan,
61 av. du Prés. Wilson,
94235 Cachan cedex, France
e-mail: {duflot,fribourg,picaro}@lsv.ens-cachan.fr

The date of receipt and acceptance will be inserted by the editor

Summary. We consider Lehmann-Rabin's randomized solution to the well-known problem of the dining philosophers. Up to now, such an analysis has always required a "fairness" assumption on the scheduling mechanism: if a philosopher is continuously hungry then he must eventually be scheduled. In contrast, we modify here the algorithm in order to get rid of the fairness assumption, and we claim that the spirit of the original algorithm is preserved. We prove that, for *any* (possibly unfair) scheduling, the modified algorithm converges: every computation reaches with probability 1 a configuration where some philosopher eats. Furthermore, we are now able to evaluate the expected time of convergence in terms of the number of transitions. We show that, for some "malicious" scheduling, this expected time is at least exponential in the number N of philosophers.

1 Introduction

Recently, due to the rising risk of traffic congestion, there has been an increasing interest in providing differentiated Internet services, departing from the traditional notion of fairness for bandwidth allocations [4,7]. This motivates reconsidering the need for fairness assumptions, classical in resource-allocation algorithms (see, *e.g.*, chapter 11 of [11]). Here we consider Lehmann-Rabin's randomized solution to a special case of resource-allocation problem: the dining philosophers.
N philosophers, $P_0, \cdots, P_{N-1}$ (where N is a parameter), are seated around a table, and variously think or try to eat by using some shared forks. The problem is to find a distributed protocol guaranteeing that some philosopher will eventually eat. A philosopher is only able to execute a step provided he is selected by a general *scheduling* mechanism. When a philosopher is scheduled, he executes exactly one action (and nothing is done by the others). Let $\mathcal{L}$ be the set of configurations, called here "legitimate", where some philosopher eats. We show here that the algorithm reaches $\mathcal{L}$ within a finite time with probability 1. In the following, we call this property *convergence*. (We will also employ the term '*progress*', which is often used in the context of dining philosophers; see, *e.g.*, [11].) Up to now, such a property has always been proved using a "fairness" assumption on the scheduling: if a philosopher is continuously hungry (i.e., trying to eat) then he must eventually be scheduled. Fairness guarantees that there exist *rounds*, intervals in which each philosopher has been scheduled at least once. It is shown in [11–13,16] that within a constant number of rounds, the probability of reaching $\mathcal{L}$ is greater than 0. It follows that the algorithm converges towards a configuration in $\mathcal{L}$ with probability 1.

In contrast, we consider here *arbitrary* schedulings, without any fairness assumption (so we do not use the notion of rounds). We modify the original Lehmann-Rabin algorithm by removing self-looping actions. We show that the new algorithm still converges towards $\mathcal{L}$ with probability 1. This is done by constructing a measure Δ over configurations that decreases with positive probability at each computation step (that does not reach $\mathcal{L}$). We thus propose a solution to the resource allocation problem for dining philosophers under *arbitrary* scheduling. We also show that the expected time of convergence, in terms of individual actions, is at least exponential in N, for some "malicious" scheduling.

⋆ This paper is a revised and extended version of a communication given by the same authors, at 2nd IFIP Int. Conf. on Theoretical Computer Science (TCS@2002).

The plan of the paper is as follows. In Section 2 we first present the kind of systems we consider and our method for proving convergence when the scheduler is arbitrary. Section 3 presents Lehmann and Rabin's randomized dining philosophers algorithm, and the changes introduced in our variant. The convergence proof of our variant without fairness, as well as a computation of expected time of convergence are given in section 4. We conclude in Section 5.

2 Theoretical Framework

2.1 Randomized Uniform Ring Systems

The notions presented here are inspired from [9] and [17]. A *randomized uniform ring system* is a triple $(N, \underset{\mathcal{R}}{\longrightarrow}, Q)$ where N is the number of processes in the system, $\underset{\mathcal{R}}{\longrightarrow}$ is a state transition algorithm, and Q is the *alphabet*, i.e. a finite set of process states. The N processes $P_1, ..., P_N$ form a ring: there is an edge between two consecutive processes, which means that P_i can observe the states q_{i-1} and q_{i+1} of P_{i-1} and P_{i+1} respectively. Calculations on indices i of processes are done modulo N. Let Q be the state set of P_i. The system is *uniform* in the sense that $\underset{\mathcal{R}}{\longrightarrow}$ and Q are common to all processes. A *configuration* is an N-tuple of process states (or letters); if the current state of process P_i is $q_i \in Q$, then the configuration of the system is $x = q_1 q_2 \cdots q_N$. We denote by X the set of all configurations, i.e., $X = Q^N$. The state transition algorithm $\underset{\mathcal{R}}{\longrightarrow}$ is given as a set $\mathcal{R}$ of rules, consisting of *deterministic* or *probabilistic* rewrite rules[1]. A deterministic rule is here of one the following forms:

- $q \to q'$
- $qr \to q'r$,
- $rq \to rq'$

where q, q', r denote states of Q.
In this paper, we only consider probabilistic rules being of of the form:

$$q \to \begin{cases} q'_0 \text{ with probability } p_0 \\ q'_1 \text{ with probability } 1 - p_0. \end{cases}$$

where q, q'_0, q'_1 denote states of Q. So, we associate to every probabilistic rule a random flip with two possible outcomes in $\{0, 1\}$ and, accordingly, two output letters q'_0, q'_1.

[1] Rewrite rules can be more general than those described here (e.g., with three letters in each side). For the sake of shortness, we just present the kind of rules needed for the philosophers problem.

The left-hand side letter q is the *old* letter of the rule, the right-hand side letter q' (with possible subscripts) being its *new* letter of the rule. For readability, the old and new letters will often be written in **bold** within rules. A rewrite rule $\mathtt{R}$ of the left-hand side $\mathbf{q}$ is *applicable* at position i of a configuration x if the i-th letter of x is q. Likewise, a rewrite rule $\mathtt{R}$ of left-hand side $\mathbf{q}r$ (resp. $r\mathbf{q}$) is *applicable* at position i if the i-th letter of x is q and the $(i+1)$-th (resp. $(i-1)$-th) letter is r.

Given a configuration x, we say that process P_i (or position i) is *enabled* if at least one rewrite rule is applicable to the i-th letter of x. Let $\mathcal{E}(x)$ be the set of indices of the enabled processes of x. We suppose henceforth that the system has *no deadlock*, i.e.: $\forall x \in X \quad \mathcal{E}(x) \neq \emptyset$.

Given x and an enabled position i of x, a *transition* leads from x to the configuration y obtained from x by changing the i-th letter of x equal to the old letter of some applicable rule, say $\mathtt{R}$, into the new letter. Such a transition is written $x \xrightarrow[\mathtt{R}]{i} y$ (or more simply $x \xrightarrow{i} y$). The probability associated to this transition is 1 if rule $\mathtt{R}$ is deterministic, or p_0 (resp. $p_1 = 1 - p_0$) if $\mathtt{R}$ is probabilistic, the new letter being q'_0 (resp. q'_1).

Without loss of understanding, we will abbreviate henceforth the randomized uniform ring system previously denoted $(N, \underset{\mathcal{R}}{\longrightarrow}, Q)$ as $\underset{\mathcal{R}}{\longrightarrow}$ (or sometimes even more simply as $\to$). We now arbitrarily fix a configuration x_0 as the initial configuration.

A *past behaviour up to step* j of $\to$ is a sequence of transitions $x_0 \xrightarrow[\mathtt{R}_{i_0}]{i_0} x_1 \xrightarrow[\mathtt{R}_{i_1}]{i_1} \cdots \xrightarrow[\mathtt{R}_{i_{j-2}}]{i_{j-2}} x_{j-1} \xrightarrow[\mathtt{R}_{i_{j-1}}]{i_{j-1}} x_j$ starting from x_0. (So i_k is the process selected at $k+1$-th step, for all $k \geq 0$.)

A *central schedule* is a function that assigns to every past behaviour $x_0 \xrightarrow[\mathtt{R}_{i_0}]{i_0} \cdots \xrightarrow[\mathtt{R}_{i_{j-1}}]{i_{j-1}} x_j$, the enabled position of x_j at which a rule will be applied at step $j+1$.[2]

For a given starting configuration x_0, a given schedule $\mathcal{S}$ and specific outcomes of the random flips F (F is an infinite sequence of elements of the set $\{0, 1\}$), we get a particular *computation of* $\to$ *under* $\mathcal{S}$, denoted $COM_{\mathcal{R}}(x_0, \mathcal{S}, F)$, which is an infinite sequence of transitions of the form $x_0 \xrightarrow[\mathtt{R}_{i_0}]{i_0} \cdots \xrightarrow[\mathtt{R}_{i_{j-1}}]{i_{j-1}} x_j \xrightarrow[\mathtt{R}_{i_j}]{i_j} \cdots$ where i_j is the process selected by $\mathcal{S}$ at $j+1$-th step. We shall use the term *finite computation* to denote a finite sequence of transitions.

[2] As usual in the dining philosophers problem [17], we only focus on *central* schedules (only one enabled position is selected at each step), but see Remark of section 3.3.

A computation C is *fair* if, in C, the state of every process P_i $(1 \leq i \leq N)$ is rewritten infinitely often. A schedule $\mathcal{S}$ is *fair* if, for every sequence F of outcomes of the random flips and every starting configuration x_0, the computation $COM_{\mathcal{R}}(x_0, \mathcal{S}, F)$ is fair.

An $\mathcal{L}$*-traversing* computation C is a computation of the form $x_0 \xrightarrow[\mathtt{R}_{i_0}]{i_0} \cdots \xrightarrow[\mathtt{R}_{i_{j-1}}]{i_{j-1}} x_j \xrightarrow[\mathtt{R}_{i_j}]{i_j} \cdots$ such that $x_j \in \mathcal{L}$ for some $j \geq 0$.

The function $COM_{\mathcal{R}}$ associates with every schedule $\mathcal{S}$ and every starting configuration x_0 a probability distribution on the space of computations, the probability Pr of a set $\mathcal{G}$ of computations being defined as the probability of the set of sequences of random flips F such that $COM_{\mathcal{R}}(x_0, \mathcal{S}, F)$ is in $\mathcal{G}$. For a more formal definition of the probabilistic space considered, based on particular computation sets called *cones* or *basic cylinder sets*, and for a precise characterization of the measurable computation sets (*i.e.* sets for which the probability is well defined), see [18] (c.f. [10]).

As pointed out in [17], given a schedule $\mathcal{S}$ and a starting configuration x_0, the set of $\mathcal{L}$-traversing computations has a well-defined probability:

$Pr(\{\ F \mid C = COM_{\mathcal{R}}(x_0, S, F)$ is $\mathcal{L}$-traversing $\})$,

which will be abbreviated in the following as

$Pr(x_0 \xrightarrow[\mathcal{R}]{\mathcal{S}}{}^{*} \mathcal{L})$.

Given a schedule $\mathcal{S}$, we are interested in proving the following *progress* (or *convergence*) property: no matter which initial configuration x_0 one starts from, the probability for a computation via $\xrightarrow[\mathcal{R}]{}$ under $\mathcal{S}$ to be $\mathcal{L}$-traversing, is 1, i.e.: $Pr(x_0 \xrightarrow[\mathcal{R}]{\mathcal{S}}{}^{*} \mathcal{L}) = 1$.

A sufficient condition for ensuring such a progress property is given in section 2.2.

Example: In [2], Beauquier, Gradinariu and Johnen present a randomized token circulation algorithm which ensures convergence towards the set of configurations with one probabilistic token whatever the schedule is. We consider the case where the number of processes is odd. The state of a process is a couple (d, p) where d is a deterministic state and p a probabilistic state, with $d, p \in \{0, 1\}$. In the following, $\overline{d}$ (resp. $\overline{p}$) denotes the complementary of d (resp. p). The transition system $\rightarrow$ is defined by:

$$(d,p)(\mathbf{d}, \overline{\mathbf{p}}) \rightarrow (d,p)(\overline{\mathbf{d}}, \overline{\mathbf{p}})$$

$$(d,p)(\mathbf{d}, \mathbf{p}) \rightarrow \begin{cases} (d,p)(\overline{\mathbf{d}}, \mathbf{p}) & \text{with probability } 1/2 \\ (d,p)(\overline{\mathbf{d}}, \overline{\mathbf{p}}) & \text{with probability } 1/2 \end{cases}$$

Given a configuration x, let (d_i, p_i) denote the state at position i in x. We say that there is a deterministic (resp. probabilistic) token at position i if $d_i = d_{i-1}$ (resp. $p_i = p_{i-1}$). The enabled positions are those with a deterministic token. Let us consider the configuration $x = (0,1)(1,0)(1,0)(0,1)(1,1)$. The only enabled position is 3 $(d_3 = d_2 = 1)$. Depending on the outcome of the random flip, we have $x \rightarrow y_1 = (0,1)(1,0)(0,0)(0,1)(1,1)$ with probability $1/2$, and $x \rightarrow y_2 = (0,1)(1,0)(0,1)(0,1)(1,1)$ with probability $1/2$. The set $\mathcal{L}$ of legitimate configurations is defined as the set of configurations with a single probabilistic token.

2.2 A Sufficient Condition for Progress

In [6], we gave a sufficient condition for ensuring progress, whatever the central schedule is. This result is restated here as follows:

Theorem 1. *Given a ring system* $\xrightarrow[\mathcal{R}]{}$ *with no deadlock, if there exist a measure* Δ *and an order* $\ll$ *such that*

Prop: $\forall x \notin \mathcal{L}\ \forall i \in \mathcal{E}(x)$

$$\exists y\ (x \xrightarrow[\mathcal{R}]{i} y \wedge (y \in \mathcal{L} \vee \Delta(y) \ll \Delta(x))),$$

then, for any central schedule $\mathcal{S}$: $\forall x \ \ Pr(x \xrightarrow[\mathcal{R}]{\mathcal{S}}{}^{*} \mathcal{L}) = 1$.

For the sake of self containment, a proof of Theorem 1 is given in Appendix A. The existential quantification on y in Prop corresponds to a "rewriting policy" for probabilistic transitions. Prop states that, whatever the selected process is, we can choose the output of the applicable probabilistic rule so that Δ decreases. Theorem 1 can be seen as a restricted version of Theorem 3.5 of [3] (*cf.* Theorem 1 of [1]). An example applying Theorem 1 to Beauquier-Gradinariu-Johnen's algorithm (see example of section 2.1) is given in appendix B.

In section 4, we will apply a variant of Theorem 1 (viz., Theorem 3) in order to prove the progress property for a variant of Lehmann-Rabin's algorithm, for all (central) schedules, even the unfair ones.

3 Randomized Dining Philosophers With and Without Fairness

3.1 The Dining Philosophers Problem

The dining philosophers problem was introduced by Dijkstra [5], and has become a paradigm for a large class of concurrency control problems. The idea is as follows: there are N philosophers sitting around a table (N is a parameter), with one fork between each pair of neighbours. A philosopher can either think (having no interaction with his neighbours) or try to eat. To do so, a philosopher needs both his left fork and his right fork. As both forks are each shared with a neighbour, a philosopher can eat only if none of his neighbours holds any of these forks. Each philosopher may address at most one shared variable through a rule. Each rule is a '*test and set*' operation, in which a philosopher reads the current value of the shared variable and assigns it a new value, which is a function of the old value and the philosopher's current internal state. The problem is to find an individual algorithm (set of rules) common to all the philosophers that guarantees the following *progress* property: as soon as one philosopher is hungry, some philosopher (not necessarily the same) will eat eventually, whatever the schedule is.

Lehmann and Rabin have shown in [17] that this problem has no deterministic, *truly distributed* (s.t. each philosopher may address only its internal state and one shared variable at a time) and *symmetric* (s.t. all philosophers start in the same state) solution. This is because there exists always a malicious schedule that selects philosophers in a round-about manner so that symmetry is always met at the end of each round (which prevents some philosopher to hold simultaneously two forks). Therefore, they incorporate random choices into the individual philosopher's algorithm, ensuring that, with probability one, the symmetry will be broken. Their solution, described in section 3.2, however assumes that the scheduling is *fair*.

3.2 Lehmann-Rabin's Algorithm

We present Lehmann-Rabin's probabilistic dining philosophers algorithm [17] along the lines of [13].

The state set of each philosopher is $Q = \{T, H, \overleftarrow{W}, \overrightarrow{W}, \overleftarrow{S}, \overrightarrow{S}, \overleftarrow{D}, \overrightarrow{D}, E, L_1, L_2\}$. The letter T represents thinking, H that a philosopher is hungry, $\overleftarrow{W}$ (resp. $\overrightarrow{W}$) that a philosopher waits in order to attempt to pick up the left (resp. right) fork next time he is scheduled, $\overleftarrow{S}$ (resp. $\overrightarrow{S}$) that he is holding only the left (resp. right) fork, $\overleftarrow{D}$ (resp. $\overrightarrow{D}$) that he will put down the left (resp. right) fork next time he is scheduled, E that he eats, L_1 that he will put down one fork (say, the right one), and L_2 the second one.

The details relating to the shared forks are omitted here. Thus, for example, if P_i is in state $\overleftarrow{S}$ or P_{i-1} is in state $\overrightarrow{S}$, it means the variable representing the shared fork (between P_i and P_{i-1}) has been set to a value 'taken'. Note that, because of the uniqueness of the shared variable addressed by a rule, a philosopher cannot go directly, e.g., from state $\overrightarrow{S}$ to H without passing by $\overrightarrow{D}$: he must discover that the *left* fork is held by his neighbour through a first operation before putting down the *right* fork on the table. In this modeling, not all configurations of Q^N are possible, as a fork can be taken by at most one philosopher. More precisely, we say that a configuration is *admissible* iff it does not contain any substring of the form $\overrightarrow{\alpha}\overleftarrow{\beta}$, with $\overrightarrow{\alpha} \in \{\overrightarrow{S}, \overrightarrow{D}, E, L_1\}$, $\overleftarrow{\beta} \in \{\overleftarrow{S}, \overleftarrow{D}, E, L_1, L_2\}$. It is easy to see that the set of admissible configurations is closed via application of any rule described below. Henceforth, we will assume that the starting configuration, and hence the subsequent ones, are admissible.

The set $\mathcal{R}$ of rewrite rules is:

`Q0`: $\mathbf{T} \to \mathbf{T}$
`Q1`: $\mathbf{T} \to \mathbf{H}$
`R0`: $\mathbf{H} \to \overleftarrow{\mathbf{W}}$ with probability 1/2
or $\overrightarrow{\mathbf{W}}$ with probability 1/2.
`R1`: $\neg\overrightarrow{hold}\ \overleftarrow{\mathbf{W}} \to \neg\overrightarrow{hold}\ \overleftarrow{\mathbf{S}}$
`R2`: $\overrightarrow{hold}\ \overleftarrow{\mathbf{W}} \to \overrightarrow{hold}\ \overleftarrow{\mathbf{W}}$
`R3`: $\overrightarrow{\mathbf{W}}\ \neg\overleftarrow{hold} \to \overrightarrow{\mathbf{S}}\ \neg\overleftarrow{hold}$
`R4`: $\overrightarrow{\mathbf{W}}\ \overleftarrow{hold} \to \overrightarrow{\mathbf{W}}\ \overleftarrow{hold}$
`R5`: $\overleftarrow{\mathbf{S}}\ \neg\overleftarrow{hold} \to \mathbf{E}\ \neg\overleftarrow{hold}$
`R6`: $\overleftarrow{\mathbf{S}}\ \overleftarrow{hold} \to \overleftarrow{\mathbf{D}}\ \overleftarrow{hold}$
`R7`: $\neg\overrightarrow{hold}\ \overrightarrow{\mathbf{S}} \to \neg\overrightarrow{hold}\ \mathbf{E}$
`R8`: $\overrightarrow{hold}\ \overrightarrow{\mathbf{S}} \to \overrightarrow{hold}\ \overrightarrow{\mathbf{D}}$
`R9`: $\overleftarrow{\mathbf{D}} \to \mathbf{H}$
`R10`: $\overrightarrow{\mathbf{D}} \to \mathbf{H}$
`R11`: $\mathbf{E} \to \mathbf{L}_1$
`R12`: $\mathbf{L}_1 \to \mathbf{L}_2$
`R13`: $\mathbf{L}_2 \to \mathbf{T}$

where $\overrightarrow{hold}$ (resp. $\overleftarrow{hold}$) denotes any state of Q corresponding to a philosopher holding his right fork (resp. left fork), *i.e.* $\overrightarrow{S}$, $\overrightarrow{D}$, E or L_1 (resp. $\overleftarrow{S}$, $\overleftarrow{D}$, E, L_1 or L_2), and $\neg\overrightarrow{hold}$ (resp. $\neg\overleftarrow{hold}$) denotes any state of the complementary set.

The rules describe the behaviour of a selected philosopher as follows: initially he thinks "repeatedly" (`Q0`); he becomes hungry (`Q1`); he decides randomly which fork to pick up first (`R0`); next he persists with his decision (`R2` or `R4`) until he finally picks it up when available (`R1` or `R3`), only putting it down later if he finds that his other fork is already held by his neighbour (`R6` followed by `R9`, or `R8` followed by `R10`); if he finds that his other fork is not held, he takes it and eats (`R5` or `R7`). After that, he leaves the eating phase (`R11`), puts down the left fork (`R12`), then the right fork (`R13`), going back to thinking phase. This behaviour is depicted on figure 1 (drawn from [15]).

The legitimate set $\mathcal{L}$ is here the set of all (admissible) configurations of Q^*EQ^*, *i.e.* the configurations in which at least one philosopher is eating.

3.3 Our Variant: Removal of Stuttering Rules

Let us observe that rule `Q0` (resp. `R2`, `R4`) is "stuttering" in the sense that the old and new letters of the rule coincide. When a selected philosopher is thinking (resp. waiting for picking up a first fork held by a neighbour), a transition that does not change the configuration may occur. This is depicted by a self-loop on state T (resp. $\overleftarrow{W}$, $\overrightarrow{W}$) in figure 1. We modify Lehmann-Rabin's algorithm mainly by removing stuttering rules `Q0`, `R2` and `R4`:

- without `Q0`, when a philosopher in state T is selected, his state always becomes H via `Q1`. States T and H then play the same role and will be merged together in the following;
- without `R2` (resp. `R4`), when a philosopher waits for a first fork that is held by a neighbour, i.e., is in state $\overleftarrow{W}$ (resp. $\overrightarrow{W}$) and his left (resp. right) neighbour in state $\overrightarrow{hold}$ (resp. $\overleftarrow{hold}$), he is no longer enabled: no rule applies to him. In such a situation, the philosopher cannot be selected anymore. Note that this differs from Lehmann-Rabin's original algorithm where every process can always be selected.

Since the state T is merged with H, the new state set Q' is $Q - \{T\}$ and rule `R13`: $L_2 \rightarrow T$ becomes `R13'`: $L_2 \rightarrow H$. The rewrite system $\mathcal{R}$ is transformed into $\mathcal{R}' = \mathcal{R} \cup \{\texttt{R13'}\} - \{\texttt{Q0}, \texttt{Q1}, \texttt{R2}, \texttt{R4}, \texttt{R13}\}$. The behaviour of a selected philosopher under $\mathcal{R}'$ is depicted on figure 2. Accordingly, the new legitimate set $\mathcal{L}'$ is (the set of admissible configurations in) $Q'^*EQ'^*$.

Discussion.

In Lehmann-Rabin's algorithm, a non-eating philosopher either thinks (state T) or tries to eat (states $\{H, W, S, D\}$). In our version of the algorithm, as the state T has been merged with H, this philosopher can only try to eat. This feature may be seen as a limitation. Actually, since we have no fairness assumption on the scheduling, a philosopher can be indefinitely ignored, thus behaving in state H as he used to do originally in state T (i.e., not trying to pick up a fork). We thus claim that our modified algorithm is similar in spirit to the original one.

The original progress property of Lehmann-Rabin's algorithm can be stated as follows:

for every *fair* (central) schedule $\mathcal{S}$ and every configuration $x \in Q^*\{H, W, S, D\}Q^*$, $\quad Pr(x \xrightarrow[\mathcal{R}]{\mathcal{S}} {}^*\mathcal{L}) = 1$.

Surprisingly, as shown in section 4, for our modified version $\mathcal{R}'$ of $\mathcal{R}$, the progress property holds with *no* fairness assumption, i.e.:

Theorem 2. *For any* arbitrary *central schedule $\mathcal{S}$ and every $x \in Q'^*$,*

$$Pr(x \xrightarrow[\mathcal{R}']{\mathcal{S}} {}^*\mathcal{L}') = 1.$$

Theorem 2 will be proven in section 4 using a variant of Theorem 1, by exhibiting an appropriate measure Δ.

Remark:

The observation done in [17] (p.340) for relaxing the assumption of *central* scheduling is independent of their assumption of fairness, hence applies also in our context:

"No two rules take place at exactly the same time; this restriction could be easily lifted to allow rules on different processes, as long as they do not address the same shared variable, to take place exactly at the same time."

4 Proof of Progress Without Fairness

We are going to exhibit a measure Δ on configurations that will characterize in some sense the "distance" of the current configuration x to $\mathcal{L}'$. We will show that, with an appropriate rewriting policy, Δ decreases at each step of computation. More precisely, we will show that, for a certain choice of the output of probabilistic rule `R0` (i.e., either $\overleftarrow{W}$ or $\overrightarrow{W}$, depending on the context of H in x):

- the application of `R0` makes Δ decrease,
- the application of any other (deterministic) rule makes Δ decrease or leads to $\mathcal{L}'$.

Symbol W denotes a letter of $\{\overleftarrow{W}, \overrightarrow{W}\}$. Likewise, S (resp. D) denotes a letter of $\{\overleftarrow{S}, \overrightarrow{S}\}$ (resp. $\{\overleftarrow{D}, \overrightarrow{D}\}$). Let $\overleftarrow{Q'} = \{H, \overleftarrow{W}, \overleftarrow{S}, \overleftarrow{D}\}$ and $\overrightarrow{Q'} = \{H, \overrightarrow{W}, \overrightarrow{S}, \overrightarrow{D}\}$. (Note that $\overleftarrow{Q'} \cap \overrightarrow{Q'} = \{H\}$.)

4.1 Ideas behind the Proof

In order to define Δ, we exploit the fact that, after a finite time, any non-legitimate configuration can be decomposed into:

- "bonds", i.e. strings of two letters in $\overleftarrow{Q'}\overrightarrow{Q'}$.
- "anti-bonds", that are, roughly speaking, strings of two letters in $\overrightarrow{Q'}\overleftarrow{Q'}$.

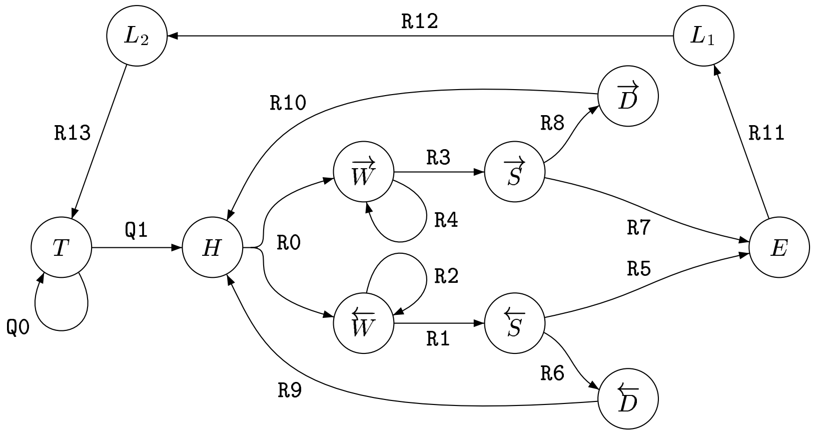

Fig. 1. Illustration of Lehmann and Rabin's algorithm.

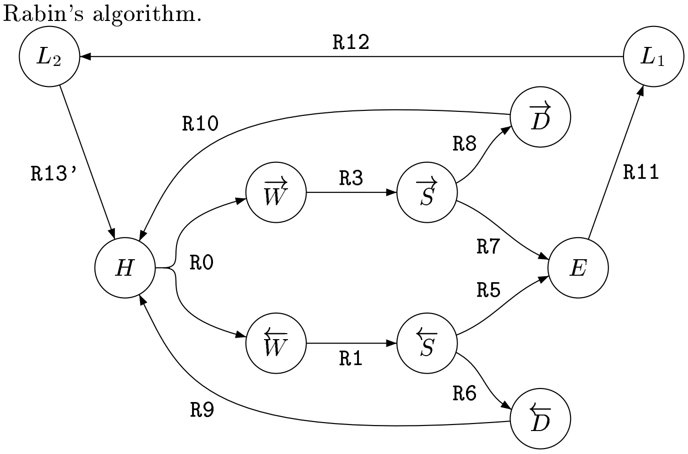

Fig. 2. Illustration of our variant algorithm.

- letters belonging to neither a bond nor an anti-bond. (These letters are in $\{W, S, D\}^*$, since every H belongs to a bond or an anti-bond; see Proposition 2 below.)

If at some point the configuration has no bond, for example $\overleftarrow{W}^N$, then, the first time an H is produced, a bond appears. Our rewriting policy aims at preserving the bonds and their position, i.e., replacing via R0: $\mathbf{H}\overrightarrow{\beta}$ into $\overleftarrow{\mathbf{W}}\overrightarrow{\beta}$, and $\overleftarrow{\alpha}\mathbf{H}$ into $\overleftarrow{\alpha}\overrightarrow{\mathbf{W}}$. With this rewriting policy, a bond never disappears once it has been created.

A bond corresponds to the situation described in [17], in which some philosopher's last random flip is left while his right neighbour's last random flip is right. We know that in this case, after a finite number of rewritings of the bond, one of the two philosophers will eat (Lemma 3, p.342, in [17]). In Lehmann-Rabin's context, the proof of progress is then almost done because, by fairness assumption on the scheduling, every bond is guaranteed to be infinitely rewritten along any (infinite) computation. Still in our context, we have to show that no infinite rewriting can occur outside bonds (i.e., at anti-bond positions or between bonds and anti-bonds). We shall use the fact that, after a bond has been initially created, every configuration is a repeated sequence of the form:

$$\overleftarrow{\alpha_1}\overrightarrow{\beta_1}\ \overrightarrow{Q'}^*\ \overrightarrow{\gamma}\overleftarrow{\delta}\ \overleftarrow{Q'}^*\ \overleftarrow{\alpha_2}\overrightarrow{\beta_2} \quad \text{or} \quad \overleftarrow{\alpha_1}\overrightarrow{\beta_1}\ \overleftarrow{\alpha_2}\overrightarrow{\beta_2}.$$

In the first case, an anti-bond $\overrightarrow{\gamma}\overleftarrow{\delta}$ lies between two consecutive bonds; in the second case, the two consecutive bonds are adjacent. We shall also use a measure Δ, defined as a 7-uple $(\Delta_1, \cdots, \Delta_7)$. Each component Δ_i is, roughly speaking, a function from Q^N in $\mathbb{N}$, which makes Δ decrease lexicographically at each step of rewriting (unless $\mathcal{L}'$ is reached). This is shown by a tedious case analysis (see section 4.4), the main cases of which are summarized below.

First consider the case where rewriting occurs at a bond position. As already noticed, thanks to our rewriting policy, Δ_1 is preserved. On the other hand, rewriting of D into H, H into W, or W into S decreases Δ_2 (because $\Delta_2(D) > \Delta_2(H) > \Delta_2(W) > \Delta_2(S)$) while rewriting of S into E yields a configuration of $\mathcal{L}'$. Therefore the rewriting of a bond always decreases Δ or yields a legitimate configuration.

Let us now sketch out why Δ also decreases when anti-bonds are rewritten. Our rewriting policy for anti-bonds aims at keeping them at the same position. For example, an anti-bond of the form $\mathbf{H}\overleftarrow{\delta}$ will be written into $\overrightarrow{\mathbf{W}}\overleftarrow{\delta}$, and $\overrightarrow{\gamma}\mathbf{H}$ into $\overrightarrow{\gamma}\overleftarrow{\mathbf{W}}$. Besides, anti-bonds are of two kinds:

- unoriented (i.e. of the form $H\overleftarrow{W}$, $\overrightarrow{W}\overleftarrow{W}$ or $\overrightarrow{W}H$), or
- leftward oriented (i.e. of the form $\{\overrightarrow{S},\overrightarrow{D}\}\overleftarrow{\delta}$) or rightward oriented (i.e. of the form $\overrightarrow{\gamma}\{\overleftarrow{S},\overleftarrow{D}\}$).

An unoriented anti-bond cannot be rewritten twice without becoming oriented (thus decreasing the opposite number Δ_4 of oriented anti-bonds). Furthermore, an oriented anti-bond of the form, say $\{\overrightarrow{S},\overrightarrow{D}\}\overleftarrow{\delta}$, when rewritten, can either:

- stay at the same position with the same orientation, which decreases Δ_6 (the sum of the coefficients of anti-bond letters),
- stay at the same position but loosing its orientation, which decreases Δ_3 (accounting for the number of $\{H,\overrightarrow{W}\}\overrightarrow{D}$ and $\overleftarrow{D}\{H,\overleftarrow{W}\}$), or
- move one position left with the same orientation, which decreases Δ_5 (accounting for the distance between oriented anti-bonds and their closest bonds).

For example, the rewriting of $\overrightarrow{\mathbf{S}}\overleftarrow{W}$ into $\overrightarrow{\mathbf{D}}\overleftarrow{W}$ decreases Δ_6 (because $\Delta_6(S) > \Delta_6(D)$). On the other hand, the rewriting of $\lambda\overrightarrow{\mathbf{D}}\overleftarrow{W}$ into $\lambda\mathbf{H}\overleftarrow{W}$ decreases Δ_3 or Δ_5 depending on the letter λ to the left of the anti-bond: if λ is $\overrightarrow{W}$, then $\overrightarrow{W}\overrightarrow{\mathbf{D}}\overleftarrow{W}$ rewrites to $\overrightarrow{W}\mathbf{H}\overleftarrow{W}$, the anti-bond becomes unoriented and Δ_3 decreases (because an occurrence of $\overrightarrow{W}\overrightarrow{D}$ disappears); if λ is $\overrightarrow{S}$, then $\overrightarrow{S}\overrightarrow{\mathbf{D}}\overleftarrow{W}$ rewrites to $\overrightarrow{S}\mathbf{H}\overleftarrow{W}$, the anti-bond becomes $\overrightarrow{S}\mathbf{H}$, one position to the left, and Δ_5 decreases. In any case, the rewriting of an anti-bond decreases Δ.

Suppose finally that rewriting occurs between bonds and anti-bonds. Recall that letters between bonds and anti-bonds are only of the form $\overleftarrow{D}$, $\overleftarrow{W}$, $\overleftarrow{S}$ or symmetrically (see Proposition 2). Rewriting of D into H creates a new bond (thus decreasing Δ_1); rewriting of W into S, or S into D decreases Δ_7 (because $\Delta_7(W) > \Delta_7(S) > \Delta_7(D)$) while rewriting of S into E yields a legitimate configuration. In any case, rewriting decreases Δ or yields a configuration of $\mathcal{L}'$.

Therefore, at any position, every step of rewriting decreases Δ unless $\mathcal{L}'$ is reached. This ends our informal explanation of why Δ always decreases.

A typical example of computation, with the associated evolution of Δ, is given in Appendix A. Note that every H of a given configuration x can belong a priori to two "overlapping" bonds of x (see section 4.2). In order to solve such ambiguities, every configuration x will be coupled with two lists: a bond list π defined from x, and an anti-bond list ψ defined from x and π. Δ will be defined for every triple (x,π,ψ). In order to prove the progress property, we will use a version of Theorem 1 reformulated as follows:

Theorem 3. *Given a ring system $\xrightarrow[\mathcal{R}']{}$ with no deadlock, if there exist a measure Δ and an order $\ll$ such that*
Prop': $\forall(x,\pi,\psi)$ *with* $x \notin \mathcal{L}', \forall i \in \mathcal{E}(x) \quad \exists(x',\pi',\psi')$
$(x \xrightarrow[\mathcal{R}']{i} x' \wedge (x' \in \mathcal{L}' \ \vee \ \Delta(x',\pi',\psi') \ll \Delta(x,\pi,\psi)))$,
then, for any central schedule $\mathcal{S}$:
$\forall x \quad Pr(x \xrightarrow[\mathcal{R}']{\mathcal{S}}{}^{*} \mathcal{L}') = 1.$

Sections 4.2 to 4.4 are devoted to the formal proof of statement Prop' (see Proposition 3). The configurations will be implicitly non-legitimate (i.e, belong to $(Q' - \{E\})^*$). For the sake of simplicity, we will also focus on configurations which do not contain any letter L_1 or L_2 (obtained when a philosopher, after eating, puts down his forks, one after another). This is not a restriction as long as no philosopher has eaten yet. We explain at the end of section 4.4 how this proof can be modified to consider also states L_1 and L_2.

4.2 Bonds and Δ_1, Δ_2, Δ_3

A *bond* in a configuration x is a substring of x made of two consecutive letters in $\overleftarrow{Q'}\overrightarrow{Q'}$.
The *index* of a bond $\overleftarrow{\alpha}\overrightarrow{\beta}$ is the position of its first letter $\overleftarrow{\alpha}$. Note that, due to letter H, two bonds may overlap: for example, in expression $\overleftarrow{W}H\overrightarrow{S}$ there are two overlapping bonds $\overleftarrow{W}H$ and $H\overrightarrow{S}$. In the following, given a configuration x, we focus on a sequence π of indices of *disjoint* bonds of x, i.e., such that $i+2 \leq j$, for all consecutive indices i and j of π. We suppose also that π is *maximal*, i.e., such that between two consecutive indices $i,j \in \pi$ there is no bond of index k with $i+2 \leq k \leq j-2$. A maximal sequence π of indices of disjoint bonds of x is called a *bond list* of x. Note that such a list is not unique. $Bond(\pi)$ is defined as the set of letters of x at position ℓ such that $\ell = i$ or $\ell = i+1$ for some $i \in \pi$.

Example: For the configuration $\overleftarrow{W}\overrightarrow{W}\overleftarrow{S}\overleftarrow{W}H\overrightarrow{D}$, there are two possible bond lists $\pi_1 = \{1,4\}$ and $\pi_2 = \{1,5\}$. Bonds are $\overleftarrow{W}\overrightarrow{W}$ and $\overleftarrow{W}H$ for π_1, and $\overleftarrow{W}\overrightarrow{W}$ and $H\overrightarrow{D}$ for π_2.

Henceforth, every non-legitimate configuration x will be provided with a bond list π. The bond list π_0 of the initial configuration x_0 is arbitrary. Given a configuration x, a bond list π of x, and a rewriting of x into x' via some rule of $\mathcal{R}'$, the bond list π' associated with x' is constructed from π as follows:

- If the rewriting changes a $H \in Bond(\pi)$ via probabilistic rule `R0`, we apply the following rewriting policy:
  - if H is the first letter of a bond of π, then H is changed into $\overleftarrow{W}$,
  - if H is the second letter of a bond of π, then H is changed into $\overrightarrow{W}$.

  Bonds of π are thus preserved, and we let: $\pi' = \pi$.
- If the rewriting creates a new bond of index k disjoint from every bond of π, then π' is $\pi \cup \{k\}$.
- In all other cases, we let $\pi' = \pi$.

We now define Δ_1, Δ_2, Δ_3 as follows:

$\underline{\Delta_1}$: Let $\Delta_1(x,\pi)$ be N minus the number of elements of π.
The *bond coefficient* is 3 for D, 2 for H, 1 for W and 0 for S. The *weight of a bond* $\overleftarrow{\alpha}\overrightarrow{\beta}$ is the sum of the bond coefficients of $\overleftarrow{\alpha}$ and $\overrightarrow{\beta}$: for example, the weight of bond HH is 4.

$\underline{\Delta_2}$: We define $\Delta_2(x,\pi)$ as the sum of the weights of all the bonds of x indexed by π.

$\underline{\Delta_3}$: The component $\Delta_3(x)$ is defined as the number of two-letter strings of the form $\overleftarrow{D}H$, $\overleftarrow{D}\overleftarrow{W}$, $H\overrightarrow{D}$ or $\overrightarrow{W}\overrightarrow{D}$ of x.

Example: In the configuration $\overleftarrow{W}\overrightarrow{W}\overleftarrow{S}\overleftarrow{W}H\overrightarrow{D}$ of the previous example, we have $\Delta_1 = 4$, $\Delta_2 = 5$ for $\pi_1 = \{1,4\}$, and $\Delta_1 = 4$, $\Delta_2 = 7$ for $\pi_2 = \{1,5\}$. In both cases, $\Delta_3 = 1$.

4.3 Anti-bonds and Δ_4, Δ_5, Δ_6

Given a configuration x and a bond list π of x, an *anti-bond* is a substring of two letters of x $\overrightarrow{\gamma}\overleftarrow{\delta} \in \overrightarrow{Q'}\overleftarrow{Q'}$ such that $\overrightarrow{\gamma} \notin Bond(\pi)$ or $\overleftarrow{\delta} \notin Bond(\pi)$. The last condition means that $\overrightarrow{\gamma}$ and $\overleftarrow{\delta}$ do not belong simultaneously to bonds of π.

The *index of an anti-bond* of x is the position of its first letter ($\overrightarrow{\gamma}$ in this description).

Consider two bonds $\overleftarrow{\alpha}\overrightarrow{\beta}$ and $\overleftarrow{\alpha}'\overrightarrow{\beta}'$ indexed by consecutive index i and i' of π. Then either:

- $\overleftarrow{\alpha}\overrightarrow{\beta}$ and $\overleftarrow{\alpha}'\overrightarrow{\beta}'$ are contiguous (i.e: $i' = i + 2$) and there is no anti-bond in between (i.e: no anti-bond indexed by j with $i + 1 \le j \le i' - 1$), or
- $\overleftarrow{\alpha}\overrightarrow{\beta}$ and $\overleftarrow{\alpha}'\overrightarrow{\beta}'$ are not contiguous (i.e: $i' \ge i + 3$).

In the latter case, it is easy to see that, between $\overleftarrow{\alpha}\overrightarrow{\beta}$ and $\overleftarrow{\alpha}'\overrightarrow{\beta}'$, there is no substring of the form $\cdots \overleftarrow{\gamma} \cdots \overrightarrow{\delta} \cdots$ with $\overleftarrow{\gamma} \in \overleftarrow{Q'}$ and $\overrightarrow{\delta} \in \overrightarrow{Q'}$. (Otherwise, there would be a disjoint bond between $\overleftarrow{\alpha}\overrightarrow{\beta}$ and $\overleftarrow{\alpha}'\overrightarrow{\beta}'$, and π would not be maximal.) Hence the substring between $\overleftarrow{\alpha}\overrightarrow{\beta}$ and $\overleftarrow{\alpha}'\overrightarrow{\beta}'$ is of the form $\overrightarrow{Q'}^*\overleftarrow{Q'}^*$ with either no H (case (H0)) or just one H (case (H1)). More precisely, the substring delimited by the two bonds is of the form:

- (H0): $\overleftarrow{\alpha}\overrightarrow{\beta}\overrightarrow{I}\overleftarrow{I}\overleftarrow{\alpha}'\overrightarrow{\beta}'$, or
- (H1): $\overleftarrow{\alpha}\overrightarrow{\beta}\overrightarrow{I}H\overleftarrow{I}\overleftarrow{\alpha}'\overrightarrow{\beta}'$,

with $\overrightarrow{I} \in \{\overrightarrow{W}, \overrightarrow{S}, \overrightarrow{D}\}^*$ and $\overleftarrow{I} \in \{\overleftarrow{W}, \overleftarrow{S}, \overleftarrow{D}\}^*$. Let $\overrightarrow{\lambda}$ and $\overleftarrow{\mu}$ be the last letter of $\overrightarrow{\beta}\overrightarrow{I}$ and the first letter of $\overleftarrow{I}\overleftarrow{\alpha}'$ respectively. Between the two bonds, there is:

- (H0): a single anti-bond, viz: $\overrightarrow{\lambda}\overleftarrow{\mu}$, or
- (H1): two overlapping anti-bonds, viz: $\overrightarrow{\lambda}H$ and $H\overleftarrow{\mu}$.

Given π, we construct a so-called *anti-bond list* ψ of x as a set of indices obtained by putting, for every couple of non-contiguous consecutive bonds indexed by π:

- the index of $\overrightarrow{\lambda}\overleftarrow{\mu}$ in case (H0),
- the index of either $\overrightarrow{\lambda}H$ or $H\overleftarrow{\mu}$ in case (H1).

Given a configuration x and a bond list π of x, an anti-bond list ψ of x, is thus a maximal set of indices j of anti-bonds of (x,π). More precisely:

Proposition 1. *Given a configuration x and a bond list π of x, an anti-bond list ψ of x, is such that, for any couple of consecutive indices $i, i' \in \pi$, either:*

- *the bonds indexed by i and i' are contiguous ($i' = i + 2$), in which case no anti-bond of ψ lies between them (i.e., no $j \in \psi$ such that $i + 1 \le j \le i' - 1$), or*
- *they are not contiguous ($i' \ge i + 3$), in which case exactly one anti-bond indexed by $j \in \psi$ lies between them (i.e., $\exists!\ j \in \psi :\ i + 1 \le j \le i' - 1$).*

Note that, in any case, the occurrence of H (if any) between $\overleftarrow{\alpha}\overrightarrow{\beta}$ and $\overleftarrow{\alpha}'\overrightarrow{\beta}'$ always belongs to an anti-bond indexed by an element of ψ. Formally:

Proposition 2. *For a given configuration $x \notin \mathcal{L}'$, a bond list π of x and an anti-bond list ψ, every H of x belongs to a bond of π or an anti-bond of ψ.*

Example: For the configuration $\overrightarrow{W}H\overleftarrow{S}\overleftarrow{D}\overrightarrow{W}\overleftarrow{W}\overleftarrow{D}$, we have one bond list $\pi = \{3,7\}$ and two possible anti-bond lists $\psi_1 = \{1,5\}$ and $\psi_2 = \{2,5\}$. Anti-bonds are $\overrightarrow{W}H$ and $\overrightarrow{W}\overleftarrow{W}$ for ψ_1, $H\overleftarrow{S}$ and $\overrightarrow{W}\overleftarrow{W}$ for ψ_2.

Henceforth, every configuration x coupled with a bond list π, will be provided with an anti-bond list ψ. The anti-bond list ψ_0 associated with the initial couple (x_0,π_0) is arbitrary. Given a couple (x,π) and an associated anti-bond list ψ, the rewriting of x into x' via probabilistic rule `R0` preserves π when a bond is rewritten, using the rewriting policy described in section 4.2. Rewriting via `R0` also preserves ψ using the following rewriting policy:

- if H is the 1st letter of an anti-bond of ψ, then H is changed into $\overrightarrow{W}$;

- if H is the 2nd letter of an anti-bond of ψ, then H is changed into $\overleftarrow{W}$.

It is easy to see that this rewriting policy is compatible with the one for bonds: if H is shared by a bond of π and an anti-bond of ψ, both rewriting policies agree for rewriting H either into $\overleftarrow{W}$ or $\overrightarrow{W}$. For example, if H is in $\overleftarrow{S}\mathbf{H}\overleftarrow{S}$, the expression rewrites to $\overleftarrow{S}\overrightarrow{\mathbf{W}}\overleftarrow{S}$. The rewriting of x into x' via the other rules transforms π into π' as explained in the previous paragraph, and ψ into ψ' where $\psi' = \psi$ except in some cases where D is replaced by H via `R9` or `R10`. (These cases are made explicit in the case analysis of the proof of Proposition 3 in section 4.4.)

We say that an anti-bond is *oriented leftwards* (resp. *oriented rightwards*) if it is of the form $\{\overrightarrow{S}, \overrightarrow{D}\}\{\overleftarrow{W}, H\}$ (resp. $\{\overrightarrow{W}, H\}\{\overleftarrow{S}, \overleftarrow{D}\}$).

Given a bond list π and an anti-bond A of index k oriented leftwards (resp. rightwards), the π-*distance* of A is $k - i$ (resp. $i - k$) where i is the index of the closest bond of π to the left (resp. right) of A.

We now define $\Delta_4, \Delta_5, \Delta_6$ as follows:

$\underline{\Delta_4}$: Let $\Delta_4(x, \psi)$ be N minus the number of oriented anti-bonds of x indexed by ψ.

$\underline{\Delta_5}$: Let $\Delta_5(x, \pi, \psi)$ be the sum of π-distances of all the oriented anti-bonds of ψ.

The *anti-bond coefficient* is 3 for H, 2 for W, 1 for S and 0 for D. The *weight of an anti-bond* $\overrightarrow{\alpha}\overleftarrow{\beta}$ is the sum of the anti-bond coefficients of $\overrightarrow{\alpha}$ and $\overleftarrow{\beta}$: for example, the weight of $H\overleftarrow{W}$ is 5.

$\underline{\Delta_6}$: Let $\Delta_6(x, \psi)$ be the sum of the weights of all the anti-bonds of x indexed by ψ.

Example: In the configuration $\overrightarrow{W}H\overleftarrow{S}\overrightarrow{D}\overrightarrow{W}\overleftarrow{W}\overleftarrow{D}$ of the previous example, we have $\Delta_4 = 7(= N)$, $\Delta_5 = 0$, $\Delta_6 = 9$ for $\psi_1 = \{1, 5\}$, and $\Delta_4 = 6$, $\Delta_5 = 1$, $\Delta_6 = 8$ for $\psi_2 = \{2, 5\}$. In ψ_1 no anti-bond is oriented. In ψ_2, the anti-bond $H\overleftarrow{S}$ is oriented rightwards. Its π-distance with bond $\overleftarrow{S}\overrightarrow{D}$ is 1.

4.4 Measure Δ and Progress Proof

The *WSD-coefficient* is 2 for W, 1 for S and 0 for D.

$\underline{\Delta_7}$: Let $\Delta_7(x)$ be the sum of the WSD-coefficients of all the letters of x distinct from H.

Example: In the configuration $\overrightarrow{W}H\overleftarrow{S}\overrightarrow{D}\overrightarrow{W}\overleftarrow{W}\overleftarrow{D}$, as there are one S and three W, we have $\Delta_7 = 7$.

Let us now define Δ:

$\underline{\Delta}$: Given a configuration $x \notin \mathcal{L}'$, a bond list π and an anti-bond list ψ, measure Δ is defined as a 7-tuple $(\Delta_1, \Delta_2, \Delta_3, \Delta_4, \Delta_5, \Delta_6, \Delta_7)$.

To prove that the measure Δ decreases at each step for our rewriting policy, we will use the following lemma:

Lemma 1. *Consider a configuration $x \notin \mathcal{L}'$, a bond list π of x, and a configuration x'such that $x \to x'$ for the bond list π' of x' constructed as described before. If Δ_1 and Δ_2 stay constant for this transition, then $\Delta_3(x') \leq \Delta_3(x)$.*

Proof. By contraposition. Given $x \notin \mathcal{L}'$, a bond list π and a configuration x' such that $x \to x'$ with $\Delta_3(x') > \Delta_3(x)$, let us show that there exists a bond list π' of x' such that $\Delta_1(x', \pi') < \Delta_1(x, \pi)$ or $\Delta_2(x', \pi') < \Delta_2(x, \pi)$.

Since $\Delta_3(x') > \Delta_3(x)$ by assumption, a pattern of the form $\{\overrightarrow{W}, H\}\overrightarrow{D}$ or $\overleftarrow{D}\{\overleftarrow{W}, H\}$ must have appeared in x' after rewriting of x. By symmetry, we can consider only the case $x' = \cdots\{\overrightarrow{W}, H\}\overrightarrow{D}\cdots$. It implies that $x = \cdots\lambda\mu\cdots$ where $\lambda, \mu \in Q'$ are such that $\lambda\mu$ rewrites to $\{\overrightarrow{W}, H\}\overrightarrow{D}$. Hence, either:

- $\lambda = \{\overrightarrow{W}, H\}$ and μ is changed into $\overrightarrow{D}$, or
- λ is changed into $\overrightarrow{W}$ and $\mu = \overrightarrow{D}$, or
- λ is changed into H and $\mu = \overrightarrow{D}$.

The 1st case is impossible, because it would mean $\mu = \overrightarrow{S}$, and a rule $\lambda\overrightarrow{S} \to \lambda\overrightarrow{D}$ can be applied only if $\lambda \in \overrightarrow{hold}$, which is not true here.

In the 2nd case, the rule applied is `R0` and $\lambda = H$ then $\lambda\mu = H\overrightarrow{D}$ and $\Delta_3(x') = \Delta_3(x)$, which contradicts our assumption.

Let us consider the 3rd case: $\lambda\mu = \lambda\overrightarrow{D}$ changed into $H\overrightarrow{D}$. This means that λ is of the form D. We have to consider two subcases: $\lambda \in Bond(\pi)$ or $\lambda \notin Bond(\pi)$. If $\lambda \in Bond(\pi)$, then we set $\pi' = \pi$ and we have $\Delta_2(x', \pi') < \Delta_2(x, \pi)$. If $\lambda \notin Bond(\pi)$, then $\mu = \overrightarrow{D} \notin Bond(\pi)$ (as it cannot be the first letter of a bond).The letter λ must be oriented rightwards: $\lambda = \overrightarrow{D}$. (Otherwise $\lambda\mu = \overleftarrow{D}\overrightarrow{D}$ would be a bond of x, disjoint from $Bond(\pi)$ since it does not contain a H.) So $\lambda\mu = \overrightarrow{D}\overrightarrow{D}$ is changed into $H\overrightarrow{D}$. A new bond $H\overrightarrow{D}$ (disjoint from π) is thus created at index, say k. We have $\pi' = \pi \cup \{k\}$, hence $\Delta_1(x', \pi') < \Delta_1(x, \pi)$.

In every case, we showed that: if $\Delta_3(x') > \Delta_3(x)$ then $\Delta_1(x', \pi') < \Delta_1(x, \pi) \vee \Delta_2(x', \pi') < \Delta_2(x, \pi)$. $\square$

By Proposition 2, any configuration can be decomposed into bonds, anti-bonds and W, S, D-letters. Using this fact, it can be shown by case analysis that, under our rewriting policy, for all (x, π, ψ), Δ decreases when x rewrites to x' and lists π' and ψ', associated to x', are constructed from π and ψ as described in sections 4.2

and 4.3. Formally, let $\ll$ be the lexicographic extension of $<$. We have:

Proposition 3. *For every $x \notin \mathcal{L}'$, every bond list π and anti-bond list ψ of x, and every position i in $\mathcal{E}(x)$, there exist a configuration x', a bond list π' and an anti-bond list ψ' of x' such that:*

$$x \xrightarrow[\mathcal{R}']{i} x' \wedge (x' \in \mathcal{L}' \ \vee \ \Delta(x', \pi', \psi') \ll \Delta(x, \pi, \psi)).$$

Proof. Consider a non-legitimate configuration x, a list π of x, and an anti-bond list ψ. Suppose that $x \xrightarrow[\mathcal{R}']{i} x'$ by applying the rewriting policy described in the previous subsections used for probabilistic rule `R0`. To prove the theorem, we need to consider only $x' \notin \mathcal{L}'$ (so rules `R5` and `R7` are not used), and $x \notin \mathcal{L}'$ (so rule `R11` is not used). In a first time, we assume that x does not contain any L_1 or L_2 (so rules `R12` and `R13'` are not used). We are going to show that, for any rule of the form $S \to D$ (i.e.: `R6`, `R8`), $D \to H$ (i.e.: `R9`, `R10`), $H \to W$ (i.e.: `R0`) or $W \to S$ (i.e.: `R1`, `R3`), we have $\Delta(x', \pi', \psi') \ll \Delta(x, \pi, \psi)$. The proof is done by a case analysis depending on the location of the letter changed by rewriting: inside a bond, inside an anti-bond (but not a bond), or anywhere else.

In the following, when we say that a component Δ_i ($2 \leq i \leq 7$) of Δ decreases, we implicitly mean that the previous components (Δ_j for $j < i$) stay constant. When changes in π or ψ are not specified, then $\pi' = \pi$ and $\psi' = \psi$. If the rewriting occurs:

1. In a bond $\overleftarrow{\alpha}\overrightarrow{\beta}$ of π:
   - rewriting via probabilistic rule $H \to W$ (`R0`) preserves Δ_1 thanks to our rewriting policy, and decreases Δ_2.
   - rewriting via $D \to H$ or $W \to S$ decreases Δ_2.
   - rewriting of $\overleftarrow{\alpha}$ via `R6`: $\overleftarrow{S} \to \overleftarrow{D}$ is impossible since the right neighbour $\overrightarrow{\beta}$ cannot be $\overleftarrow{hold}$. Symmetrically, rewriting of $\overrightarrow{\beta}$ via `R8`: $\overrightarrow{S} \to \overrightarrow{D}$ is impossible because the left neighbour $\overleftarrow{\alpha}$ cannot be $\overrightarrow{hold}$.
2. In an anti-bond A of ψ, indexed by i:
   (a) If A is a π-disjoint anti-bond $\overrightarrow{\gamma}\overleftarrow{\delta}$ ($\overrightarrow{\gamma} \in \{\overrightarrow{W}, \overrightarrow{S}, \overrightarrow{D}\}$ and $\overleftarrow{\delta} \in \{H, \overleftarrow{W}\}$, or $\overrightarrow{\gamma} \in \{H, \overrightarrow{W}\}$ and $\overleftarrow{\delta} \in \{\overleftarrow{W}, \overleftarrow{S}, \overleftarrow{D}\}$), then Δ_1 and Δ_2 stay constant, and from lemma 1, Δ_3 cannot increase. We have several subcases.
   - $A = H\overleftarrow{W}$ and H is rewritten: with our rewriting policy we obtain $\overrightarrow{W}\overleftarrow{W}$ and Δ_6 decreases.
   - $A = H\overleftarrow{W}$ and $\overleftarrow{W}$ is rewritten: we obtain $H\overleftarrow{S}$ and Δ_4 decreases.
   - $A = \overrightarrow{W}\overleftarrow{W}$: we obtain $\overrightarrow{S}\overleftarrow{W}$ or $\overrightarrow{W}\overleftarrow{S}$, and Δ_4 decreases.
   - $A = \overrightarrow{S}\overleftarrow{W}$: since $\overleftarrow{W}$ cannot be rewritten, we obtain $\overrightarrow{D}\overleftarrow{W}$ and Δ_6 decreases.
   - $A = H\overleftarrow{hold}$ and H is rewritten: with our rewriting policy we obtain $\overrightarrow{W}\overleftarrow{hold}$, and Δ_6 decreases.
   - $A = H\overleftarrow{S}$ and $\overleftarrow{S}$ is rewritten: Δ_6 decreases.
   - $A = H\overleftarrow{D}$ and $\overleftarrow{D}$ is rewritten: a new disjoint bond HH is created with index i. We let $\pi' = \pi \cup \{i\}$, thus decreasing Δ_1. ψ' is an anti-bond list associated with (x', π'), chosen arbitrarily.
   - $A = \overrightarrow{D}\overleftarrow{W}$: A is an anti-bond oriented leftwards. Since $\overleftarrow{W}$ cannot be rewritten, A is rewritten via $\overrightarrow{D} \to H$ into $H\overleftarrow{W}$. The left neighbour of A is a letter $\overrightarrow{\lambda} \in \overrightarrow{Q'}$. (Otherwise $\overrightarrow{D}$ would belong to a bond, and A would not be π-disjoint.) There are two possibilities:
     – If $\overrightarrow{\lambda} \in \{H, \overrightarrow{W}\}$, then Δ_3 decreases.
     – If $\overrightarrow{\lambda} = \overrightarrow{hold}$, the substring $\overrightarrow{\lambda}A = \overrightarrow{hold}\overrightarrow{D}\overleftarrow{W}$ rewrites to $\overrightarrow{hold}H\overleftarrow{W}$. There are now two overlapping anti-bonds $\overrightarrow{hold}H$ and $H\overleftarrow{W}$ at position $i-1$ and i. Let us select $\overrightarrow{hold}H$ for the new anti-bond list, i.e. let $\psi' = \psi - \{i\} \cup \{i-1\}$ and $\pi' = \pi$. The new anti-bond $\overrightarrow{hold}H$ is still oriented leftwards, but its distance to the closest bond of $\pi = \pi'$ to the left, is smaller than the distance of $A = \overrightarrow{D}\overleftarrow{W}$. So Δ_4 stays constant and Δ_5 decreases by 1.
   - Cases $A = \overrightarrow{W}H, \overrightarrow{W}\overleftarrow{S}, \overrightarrow{W}\overleftarrow{D}, \overrightarrow{S}H$ and $\overrightarrow{D}H$ are symmetrical to cases previously treated, hence omitted.

   (b) If A is not π-disjoint, then it is of the form $\overrightarrow{\gamma}\overleftarrow{\alpha}$ with $\overleftarrow{\alpha} \in Bond(\pi)$, or $\overrightarrow{\beta}\overleftarrow{\delta}$ with $\overrightarrow{\beta} \in Bond(\pi)$. Suppose $A = \overrightarrow{\gamma}\overleftarrow{\alpha}$. (The other case is symmetrical and omitted.) Since the rewriting of $\overleftarrow{\alpha}$ has already been studied in case 1, we only consider the rewriting of $\overrightarrow{\gamma}$. For the same reasons as in case 2(b), as long as we do not change π and ψ, Δ_1 and Δ_2 stay constant. We have several subcases.
   - $\overrightarrow{\gamma} = H$: with our rewriting policy, it rewrites to $\overrightarrow{W}$. Thus $A = H\overleftarrow{\alpha}$ becomes $A' = \overrightarrow{W}\overleftarrow{\alpha}$. Let $\pi' = \pi$ and $\psi' = \psi$. If A is non-oriented, then so is A'. If A is oriented of distance d, then A' is oriented in the same direction with same distance d. So Δ_4 and Δ_5 stay constant while Δ_6 decreases.
   - $\overrightarrow{\gamma} = \overrightarrow{W}$: the rewriting is possible only if $\overleftarrow{\alpha}$ is in $\{H, \overleftarrow{W}\}$. Thus $A = \overrightarrow{W}\overleftarrow{\alpha}$ becomes $A' = \overrightarrow{S}\overleftarrow{\alpha}$. Unlike A, A' is oriented, hence Δ_4 decreases.

- $\overrightarrow{\gamma} = \overrightarrow{S}$: the anti-bond becomes $\overrightarrow{D}\overleftarrow{\alpha}$ and Δ_6 decreases.
- $\overrightarrow{\gamma} = \overrightarrow{D}$: then the case is similar to the rewriting $\overrightarrow{D} \to H$ considered for the π-disjoint anti-bond (case 2(a)):
  - If the left neighbour $\overrightarrow{\lambda}$ of A is in $\{H, \overrightarrow{W}\}$, then Δ_3 decreases.
  - If $\overrightarrow{\lambda} = \overrightarrow{hold}$, then we can change ψ into ψ' so that Δ_5 decreases.

3. Outside bonds and anti-bonds:
   in this case, either a bond is created (Δ_1 decreases) or $\Delta_1, \Delta_2, \Delta_4, \Delta_5$ and Δ_6 stay constant while Δ_3 or Δ_7 decreases (by Lemma 1, Δ_3 cannot increase when Δ_1 and Δ_2 stay constant).
   More precisely, let λ be the changed letter and i its position. By proposition 2, λ belongs to $\{W, S, D\}$. We have the following subcases.
   - $\lambda = \overrightarrow{W}$: rewriting via $\overrightarrow{W} \to \overrightarrow{S}$ decreases either Δ_3 if the right neighbour of λ is $\overrightarrow{D}$, or Δ_7 otherwise.
   - $\lambda = \overrightarrow{S}$: rewriting via $\overrightarrow{S} \to \overrightarrow{D}$ decreases Δ_7.
   - $\lambda = \overrightarrow{D}$: since λ belongs to neither a bond of π nor an anti-bond of ψ, its right neighbour μ must belong to $\overrightarrow{Q'}$. (Otherwise, $\lambda\mu$ would be of the form $\overrightarrow{D}\overleftarrow{\mu}$, i.e. an anti-bond of ψ.) Rewriting λ via $\overrightarrow{D} \to H$ thus yields $H\mu$ with $\mu \in \overrightarrow{Q'}$. This is a new disjoint bond of index i. We let $\pi' = \pi \cup \{i\}$, thus decreasing Δ_1. ψ' is an anti-bond list associated with (x', π'), chosen arbitrarily.
   - $\lambda = \overleftarrow{W}, \overleftarrow{S}$ or $\overleftarrow{D}$: these cases are symmetrical to the previous ones.

This ends the proof under the assumption that configuration x did not contain any letter L_1 or L_2. The proof can easily be extended to take into account letters L_1 and L_2, as follows:

- The set $\overrightarrow{Q'}$ is augmented with the letter L_2, which corresponds to a philosopher about to put down the right fork. Definitions of bond and anti-bond are preserved as far as we consider the new set $\overrightarrow{Q'}$.
- The definition of oriented anti-bond now includes substrings of the form $L_2\overleftarrow{\delta}$.
- As a letter L_1 corresponds to a philosopher holding both forks, an anti-bond can now lie between two letters L_1 or between one letter L_1 and a bond. It can also overlap with a letter L_1. The π-distance of an anti-bond oriented leftwards (resp. rightwards) is now the minimum between the distance to the closest bond to its left (resp. right) and the distance to the closest L_1 to its left (resp. right).

The measure Δ is changed into a measure Δ' with $\Delta' = (\Delta'_1, \Delta'_2, \Delta'_3, \Delta_4, \Delta_5, \Delta'_6, \Delta'_7)$. Components 2, 4 and 5 are not changed as far as we consider the new definitions for oriented anti-bonds and π-distance.

- $\Delta'_1(x) = \Delta_1(x)+$ number of letters L_1 in x.
- $\Delta'_2(x, \pi)$ is the sum of the weights of the bonds of x indexed by π, with a bond coefficient 3 for L_2.
- $\Delta'_3(x)$ counts the number of two-letter strings of the form $\overleftarrow{D}H$, $\overleftarrow{D}\overleftarrow{W}$, $H\overrightarrow{D}$, $\overrightarrow{W}\overrightarrow{D}$, HL_2 or $\overrightarrow{W}L_2$ of x.
- $\Delta'_6(x, \psi)$ the sum of the weights of all the anti-bonds of x indexed by ψ. The anti-bond coefficient is 0 for L_2 and 1 for L_1.
- $\Delta'_7(x)$ is the sum of the WSD-coefficients of all the letters of x distinct from H and L_1. The WSD-coefficient of L_2 is 0.

□

Theorem 2 (progress property) then follows from Proposition 3 and Theorem 3.

4.5 Expected Time of Convergence

In traditional approaches (see e.g. [11]) the time is measured in terms of rounds (intervals in which each process has been scheduled at least once). The time is never evaluated as a number of transitions.

With our approach, we do not make any assumption on this round time. We evaluate the expected time of convergence as a number of transitions. It turns out that, for some "malicious" scheduler such a time can be "very" long.

Let us show on a example that the expected time of convergence is at least exponential in the number N of processes for some "malicious" scheduler. We exhibit a scheduler which, starting from the uniform configuration $x_0 = \overrightarrow{S}^N$, goes to $x_1 = \overrightarrow{S}^{N-2}\overleftarrow{W}\overleftarrow{S}$ within a constant expected time. Then, from x_1, it can stay in the set of configurations $\{\overrightarrow{S}^i\overleftarrow{W}\overleftarrow{S}^j | i+j+1 = N\} \cup \{\overrightarrow{S}^i\overrightarrow{W}\overleftarrow{S}^j | i+j+1 = N\}$ during an expected time exponential in N.

Consider $x_0 = \overrightarrow{S}^N$ as a starting configuration. Let us first show that the scheduler may reach the configuration $x_1 = \overrightarrow{S}^{N-2}\overleftarrow{W}\overleftarrow{S}$ in a finite amount of time: It selects a $\overrightarrow{S}$ and applies `R8.R10.R0` to obtain $\overrightarrow{S}^{N-1}\overrightarrow{W}$ or $\overrightarrow{S}^{N-1}\overleftarrow{W}$. In the first case, it applies `R3` and goes on. In the second case, it selects the last $\overrightarrow{S}$ and applies `R8.R10.R0` to obtain $\overrightarrow{S}^{N-2}\overrightarrow{W}\overleftarrow{W}$ or $\overrightarrow{S}^{N-2}\overleftarrow{W}\overleftarrow{W}$. In the first case, it applies `R3` to $\overrightarrow{W}$ and begins again. In the second case, it applies `R2` to the right $\overleftarrow{W}$. The expected time E of going from x_0 to x_1 may be easily computed: $E = 15$.

Now, we describe two possible choices of the scheduler on a configuration of the form $\overrightarrow{S}^i\overleftarrow{W}\overleftarrow{S}^j$ with $i \geq 2$ and $i+j+1 = N$. These two choices are the application of

four consecutive rules, one of which is a probabilistic one. The three first rules are the same: select the rightmost $\overrightarrow{S}$ and applies R8.R10.R0. This yields: $\overrightarrow{S}^{i-1}\overleftarrow{W}\overleftarrow{W}\overleftarrow{S}^{j}$ or $\overrightarrow{S}^{i-1}\overrightarrow{W}\overleftarrow{W}\overleftarrow{S}^{j}$.
From $\overrightarrow{S}^{i-1}\overleftarrow{W}\overleftarrow{W}\overleftarrow{S}^{j}$, apply R1 to the rightmost $\overleftarrow{W}$ to get $\overrightarrow{S}^{i-1}\overleftarrow{W}\overleftarrow{S}^{j+1}$. From $\overrightarrow{S}^{i-1}\overrightarrow{W}\overleftarrow{W}\overleftarrow{S}^{j}$, the scheduler can choose $\overrightarrow{W}$ or $\overleftarrow{W}$, which leads to the two cases:

(A) The scheduler applies R1 to $\overleftarrow{W}$ and yields $\overrightarrow{S}^{i-1}\overrightarrow{W}\overleftarrow{S}^{j+1}$,
(B) The scheduler applies R3 to $\overrightarrow{W}$ and yields $\overrightarrow{S}^{i}\overleftarrow{W}\overleftarrow{S}^{j}$.

Symmetrically, using R6.R9.R0, on a configuration $\overrightarrow{S}^{i}\overrightarrow{W}\overleftarrow{S}^{j}$ (with $j \geq 2$ and $i+j+1 = N$), one goes to $\overrightarrow{S}^{i}\overrightarrow{W}\overleftarrow{W}\overleftarrow{S}^{j-1}$ or $\overrightarrow{S}^{i}\overrightarrow{W}\overrightarrow{W}\overleftarrow{S}^{j-1}$.
From $\overrightarrow{S}^{i}\overrightarrow{W}\overrightarrow{W}\overleftarrow{S}^{j-1}$, apply R3 to the leftmost $\overrightarrow{W}$, to get $\overrightarrow{S}^{i+1}\overrightarrow{W}\overleftarrow{S}^{j-1}$. From $\overrightarrow{S}^{i}\overrightarrow{W}\overleftarrow{W}\overleftarrow{S}^{j-1}$, the scheduler can choose $\overrightarrow{W}$ or $\overleftarrow{W}$, which leads to the two cases:

(A') The scheduler applies R3 to $\overrightarrow{W}$ and yields $\overrightarrow{S}^{i+1}\overleftarrow{W}\overleftarrow{S}^{j-1}$.
(B') The scheduler applies R1 to $\overleftarrow{W}$ and yields $\overrightarrow{S}^{i}\overrightarrow{W}\overleftarrow{S}^{j}$.

The scheduler can iterate such an application of four consecutive rules until the system reaches an "end" configuration, i.e, a configuration of the form:
$x_{end} = \overrightarrow{S}^{N-2}\overrightarrow{W}\overleftarrow{S}$ or $y_{end} = \overrightarrow{S}\overleftarrow{W}\overleftarrow{S}^{N-2}$.

Let us consider the following "malicious" scheduler: From $\overrightarrow{S}^{i}\overleftarrow{W}\overleftarrow{S}^{j}$ (with $i \geq 2$ and $i+j+1 = N$), it chooses (A) or (B) according to the compared values of i and j. Precisely:

- it chooses (A) if $j \geq i$,
- it chooses (B) if $j < i$.

Symmetrically, from $\overrightarrow{S}^{i}\overrightarrow{W}\overleftarrow{S}^{j}$ (with $j \geq 2$ and $i+j+1 = N$):

- it chooses (A') if $i \geq j$,
- it chooses (B') if $i < j$.

For this scheduler, let us now compute the expected time for reaching x_{end} or y_{end}, starting from a configuration $\overrightarrow{S}^{i}\overrightarrow{W}\overleftarrow{S}^{j}$ (resp. $\overrightarrow{S}^{i}\overleftarrow{W}\overleftarrow{S}^{j}$), with $j \geq 2$ (resp. $i \geq 2$) and $i+j+1 = N$. Let us abbreviate this expected time as $E[\overrightarrow{i}]$ (resp. $E[\overleftarrow{i}]$). We have:

$E[\overrightarrow{i}] = 4 + 1/2\ E[\overrightarrow{i+1}] + 1/2\ E[\overleftarrow{i+1}]$, for $1 \leq i < N-2$ and $2i \geq N-1$.
$E[\overrightarrow{i}] = 4 + 1/2\ E[\overrightarrow{i+1}] + 1/2\ E[\overrightarrow{i}]$, for $1 \leq i < N-2$ and $2i < N-1$.
$E[\overleftarrow{i}] = 4 + 1/2\ E[\overrightarrow{i-1}] + 1/2\ E[\overleftarrow{i-1}]$, for $2 \leq i$ and $2i \leq N-1$.
$E[\overleftarrow{i}] = 4 + 1/2\ E[\overleftarrow{i-1}] + 1/2\ E[\overleftarrow{i}]$, for $2 \leq i$ and $2i > N-1$.
$E[\overrightarrow{N-2}] = 0$.
$E[\overleftarrow{1}] = 0$.

We then solve this linear system. The symmetry shows that $E[\overrightarrow{i}] = E[\overleftarrow{N-1-i}]$, for all $1 \leq i < N-1$. Let m be the integral part of $N/2 - 1$. The result is:

$E[\overrightarrow{i}] = 8(2^{N-3-m}(2m-N+6)-i-2)$, for $1 \leq i \leq m$.
$E[\overrightarrow{i}] = 8((2m-N+6)(2^{N-3-m}-2^{i-m-1})-N+i+2)$ for $m+1 \leq i < N-1$.

In particular, the time to go from x_1 to x_{end} (hence x_0 to x_{end}) is
$E[\overleftarrow{N-2}] = E[\overrightarrow{1}] = 8(2^{N-3-m}(2m-N+6)-3) \geq 2^{N/2}$.
This shows that we can stay out of $\mathcal{L}'$ an exponential expected number of steps, hence the upper bound for the expected time of convergence for a general scheduler is at least exponential.

5 Final Remarks

We have shown in this paper that a modified version of Lehmann-Rabin's algorithm always converges, whatever the (possibly unfair) schedule is. We claim that our modified algorithm preserves the spirit of the original one.

With our approach, we are also able to evaluate the expected time of convergence as a number of transitions. Alternatively, this time can be understood as the number of non-stuttering transitions of the original Lehmann-Rabin's algorithm. We have shown that this time may be exponential for some malicious schedule, a point that could not be seen when the expected time was computed as a number of rounds.

We recently learned that Catuscia Palamidessi and Mihaela Herescu have independently come out with the same variant of the dining philosophers without fairness, and used it to prove the possibility of encoding the π-calculus with mixed choice into the probabilistic asynchronous π-calculus [14,8]. This confirms our view that the algorithm presented here is a natural and useful variant of Lehmann-Rabin's algorithm.

Acknowledgements.
We are most grateful to Catuscia Palamidessi and anonymous referees for their helpful comments.

References

1. J. Beauquier, J. Durand-Lose, M. Gradinariu, and C. Johnen. Token based self-stabilizing uniform algorithms. *Journal of Parallel and Distributed Computing*, 62(5):899–921, 2002.

2. J. Beauquier, M. Gradinariu, and C. Johnen. Memory space requirements for self-stabilizing leader election protocols. In *Proc. 18th Annual ACM Symposium on Principles of Distributed Computing (PODC'99)*, pages 199–208. ACM Press, 1999.
3. J. Beauquier, M. Gradinariu, and C. Johnen. Randomized self-stabilizing and space optimal leader election under arbitrary scheduler on rings. Technical Report 1225, L.R.I., Orsay, France, Sept. 1999.
4. T. Bonald and L. Massoulié. Impact of fairness on internet performance. In *Proc. of Joint Int. Conf. on Measurements and Modeling of Computer Systems (SIGMETRICS/Performance 2001)*, pages 82–91, 2001.
5. E. W. Dijkstra. Hierarchical ordering of sequential processes. In *Operating Systems Techniques*, pages 72–93. Academic Press, 1972.
6. M. Duflot, L. Fribourg, and C. Picaronny. Randomized finite-state distributed algorithms as Markov chains. In *Proc. 15th Int. Conf. on Distributed Computing (DISC'01)*, volume 2180 of *LNCS*, pages 240–254. Springer, 2001.
7. P. Gevros, F. Risso, and P. Kirstein. Analysis of a method for differential TCP service. In *Proc. 4th Symposium on Global Internet (GLOBECOM'99)*, 1999.
8. O. M. Herescu. *The probabilistic asynchronous Pi-calculus.* PhD thesis, Department of Computer Science and Engineering, Pennsylvania State university, Dec. 2002.
9. H. Kakugawa and M. Yamashita. Uniform and self-stabilizing token rings allowing unfair daemon. *IEEE Trans. Parallel and Distributed Systems*, 8(2):154–163, 1997.
10. J. G. Kemeny, J. L. Snell, and A. W. Knapp. *Denumerable Markov Chains.* D. van Nostrand Co., 1966.
11. N. A. Lynch. *Distributed Algorithms.* Morgan Kaufmann Publishers, 1996.
12. N. A. Lynch, I. Saias, and R. Segala. Proving time bounds for randomized distributed algorithms. In *Proc. 13th Annual ACM Symposium on Principles of Distributed Computing (PODC'94)*, pages 314–323. ACM Press, 1994.
13. A. K. McIver. Quantitative program logic and efficiency in probabilistic distributed algorithms. Technical report, Computing Laboratory, Oxford University, UK, 1998. (Extended version of "Quantitative program logic and performance in probabilistic distributed algorithms", Proc. of 5th Int AMAST Workshop, ARTS '99).
14. C. Palamidessi and O. M. Herescu. A randomized distributed encoding of the pi-calculus with mixed choice. In *Proc. 2nd IFIP Int. Conf. on Theoretical Computer Science (TCS@02)*, volume 223 of *IFIP Conference Proceedings*, pages 537–549. Kluwer Academic, 2002.
15. A. Pnueli and L. D. Zuck. Verification of multiprocess probabilistic protocols. *Distributed Computing*, 1(1):53–72, 1986.
16. A. Pogosyants and R. Segala. Formal verification of timed properties for randomized distributed algorithms. In *Proc. 14th Annual ACM Symposium on Principles of Distributed Computing (PODC'95)*, pages 174–183. ACM Press, 1995.
17. M. O. Rabin and D. J. Lehmann. *The advantages of free choice : a symetric and fully distributed solution for the dining philosophers problem. In "A Classical Mind: Essays in Honour of C.A.R. Hoare"*, chapter 20, pages 333–352. Prentice Hall, 1994. An extended abstract appeared in the Proc. 8th Annual ACM Symposium on Principles of Programming Languages(POPL'81), p133–138.
18. R. Segala and N. A. Lynch. Probabilistic simulations for probabilistic processes. *Nordic Journal of Computing*, 2(2):250–273, 1995.

Appendix A: Proof of Theorem 1

We say that a computation C *traverses* $\mathcal{L}$ *within* m *steps* if C is a computation of the form
$x_0 \xrightarrow[\mathrm{R}_{i_0}]{i_0} \cdots \xrightarrow[\mathrm{R}_{i_{j-1}}]{i_{j-1}} x_j \xrightarrow[\mathrm{R}_{i_j}]{i_j} \cdots$ such that $x_j \in \mathcal{L}$ for some $0 \leq j \leq m$.

Given a central schedule $\mathcal{S}$, a starting configuration x_0 and $m \geq 0$, the event of being a computation traversing $\mathcal{L}$ within m steps has a well-defined probability:

$Pr(\{\ F \mid C = COM_{\mathcal{R}}(x_0, S, F)$ traverses $\mathcal{L}$ within m steps $\})$, abbreviated as $Pr(x_0 \xrightarrow[\mathcal{R}]{\mathcal{S}}{}^{\leq m} \mathcal{L})$.

Let us now show that, when Prop holds,
$lim_{m\to\infty}\ Pr(x \xrightarrow[\mathcal{R}]{\mathcal{S}}{}^{\leq m} \mathcal{L}) = 1$ (hence $Pr(x \xrightarrow[\mathcal{R}]{\mathcal{S}}{}^{*} \mathcal{L}) = 1$).

Lemma 2. *Consider a system* $\xrightarrow[\mathcal{R}]{}$ *with no deadlock, a measure* Δ *and an order* $\ll$ *such that:*
Prop: $\forall x \notin \mathcal{L}\ \forall i \in \mathcal{E}(x)$
$\quad \exists y\ (x \xrightarrow[\mathcal{R}]{i} y \wedge (y \in \mathcal{L} \vee \Delta(y) \ll \Delta(x)))$.
Then there exists an integer $M > 0$ *and a probability* $p > 0$ *such that, for any central schedule* $\mathcal{S}$*:*
$$\forall x \quad Pr(x \xrightarrow[\mathcal{R}]{\mathcal{S}}{}^{\leq M} \mathcal{L}) \geq p.$$

Proof. Let M be the number of elements of X (i.e., $M = |X| = |Q|^N$). Let q be the minimum probability associated with a probabilistic rule (for example, $q = 1/2$ for Beauquier-Gradinariu-Johnen's and Lehmann-Rabin's algorithms). Given a central schedule $\mathcal{S}$ and a starting configuration x_0, consider now a computation C under $\mathcal{S}$ of the form $x_0 \xrightarrow[\mathcal{R}]{} x_1 \xrightarrow[\mathcal{R}]{} \cdots$ such that: for every $k \geq 0$, if C has not traversed $\mathcal{L}$ at step k (i.e., $x_0 \notin \mathcal{L}, \cdots, x_k \notin \mathcal{L}$), then $x_{k+1} \in \mathcal{L}$ or $\Delta(x_{k+1}) \ll \Delta(x_k)$.

Note that, since we assume Prop, such a computation C always exists. By the pigeonhole principle, among the first $M+1$ configurations $x_0, \cdots, x_M$ of C, there are necessarily two elements x_i and x_j which are identical. Therefore C traverses $\mathcal{L}$ within M steps because, otherwise, one would have: $\Delta(x_i) \ll \Delta(x_{i+1}) \ll \cdots \ll \Delta(x_j)$, which is impossible since $x_i = x_j$. On the other hand, each step of computation has a probability greater than or equal to q. Hence the probability of the finite computation consisting of the first M steps of C is no less than q^M. It follows that the probability for all the computations starting from x_0 under $\mathcal{S}$ to traverse $\mathcal{L}$ is no less than $p = q^M$. So for any $\mathcal{S}$ and x_0, $Pr(x_0 \xrightarrow[\mathcal{R}]{\mathcal{S}}{}^{\leq M}\mathcal{L}) \geq p$. □

Proof of Theorem 1:
Let us consider a central schedule $\mathcal{S}$, arbitrary but given. Since the system $\xrightarrow[\mathcal{R}]{}$ satisfies Prop by assumption, we have, by Lemma 2: $\forall x \in X,\ Pr(x \xrightarrow[\mathcal{R}]{\mathcal{S}}{}^{\leq M}\mathcal{L}) \geq p$. So, for any starting configuration x_0, the probability of not traversing $\mathcal{L}$ from x_0 within M transitions is less than $1-p$. We can apply iteratively Prop and Lemma 2, to show that the probability of not traversing $\mathcal{L}$ within $2M$ transitions is less than $(1-p)^2$, and so on. The probability of not traversing $\mathcal{L}$ from x_0 under $\mathcal{S}$ after m transitions tends to 0 as m tends to ∞. Alternatively: $\forall x \in X,\ lim_{m\to\infty}\ Pr(x \xrightarrow[\mathcal{R}]{\mathcal{S}}{}^{\leq m}\mathcal{L}) = 1$. It follows that, for any central schedule $\mathcal{S}$: $\quad \forall x \quad Pr(x \xrightarrow[\mathcal{R}]{\mathcal{S}}{}^{*}\mathcal{L}) = 1$. □

Appendice B: Convergence of Beauquier-Gradinariu-Johnen's Algorithm

Let us consider Beauquier-Gradinariu-Johnen's algorithm presented as an example in section 2.1, with a central and arbitrary scheduling, and an odd number of processes. First observe that the system has no deadlock: as the number of processes is odd, there exists always at least one deterministic (resp. probabilistic) token in each configuration, and a process is enabled when it holds a deterministic token. We consider the measure Δ that maps any configuration x to the triple (φ, D_p, D_d) where:

- φ is the number of probabilistic tokens of x
- D_p is the minimal distance between two probabilistic tokens of x
- D_d is constructed as follows. Let $T = \{t_1, t_2, ..., t_\varphi\}$ be the set of indices of processes holding a probabilistic token (*i.e.* $p_{t_k} = p_{t_k - 1}$ for $1 \leq k \leq \varphi$). Let T' be the set of indices in T such that the distance between a probabilistic token of index t_k and the closest probabilistic token to its right (of index t_{k+1}) is minimal ($t_{k+1} - t_k = D_p$ modulo N). Then D_d is the sum over all indices i of deterministic tokens, of the distance d_i between this deterministic token, and the closest index of T' to its right. Formally:

$$D_d = \sum_{\substack{i/\text{ process } i \text{ holds}\\ \text{a deterministic token}}} \min_{j \in T'}((j-i) \text{ modulo } N)$$

These three components represent the important characteristics of a configuration: first the number of probabilistic tokens (if $\varphi = 1$ then the configuration is legitimate), then the minimal distance between probabilistic tokens (if they are close, they are more likely to collide and φ is more likely to decrease), and last the distance from deterministic tokens to probabilistic tokens that can decrease D_p. For example, in the configuration $x = (0,0)(1,0)(1,1)(0,1)(1,0)(0,0)(1,1)$ with $N = 7$, there is one deterministic token at position 3, and the set T of indices of probabilistic tokens is $\{2,4,6\}$. This is easier to see if we decompose the configuration x into a deterministic configuration 0110101 of its deterministic states and a probabilistic configuration 0011001. The minimal distance D_p is 2 and $T' = \{2,4\}$. As there is just one deterministic token at position 3, $D_d = \min_{j\in\{2,4\}}((j-3)$ modulo $N) = 1$. Let us show that, for every $x \notin \mathcal{L}$, i.e. x with at least two probabilistic tokens, and every enabled position i of x, there exists y such that $x \xrightarrow{i} y$ with $\Delta(y) \ll \Delta(x)$, where $\ll$ is the lexicographic order.
There are two cases:

- If $p_i = \overline{p_{i-1}}$, there is no probabilistic token at position i and we apply the first rule.
  - If there are two consecutive deterministic tokens at position i and $i+1$ (for example the pattern 000) then they collide (the pattern becomes 010) and both disappear. Then the two corresponding distances are removed from the sum D_d which decreases.
  - If there are no consecutive tokens at these positions then, as there is no probabilistic token at position i ($p_i = \overline{p_{i-1}}$), we have $d_i \geq 1$ and D_d decreases by 1.
- If $p_i = p_{i-1}$ then there is a probabilistic token at position i and we choose the output of the probabilistic transition according to the following rewriting policy:
  - If i belongs to T', then we change the state p_i into $\overline{p_i}$. The probabilistic token thus moves to its right and either D_p decreases, or two tokens collide (φ decreases).
  - If i does not belong to T', then $d_i \geq 1$. We keep the probabilistic state p_i unchanged and d_i decreases by one (so does D_d).

$$
\begin{array}{l}
\overleftarrow{W}\underset{\Delta_5=3}{\underline{\overrightarrow{W}\overleftarrow{\mathbf{D}}}}\overleftarrow{D}\overleftarrow{D}\overrightarrow{S}
\xrightarrow{\Delta_5}
\overleftarrow{W}\overrightarrow{W}\underset{\Delta_5=2}{\underline{\mathbf{H}\overleftarrow{D}}}\overleftarrow{D}\overrightarrow{S}
\xrightarrow{\Delta_6}
\overleftarrow{W}\overrightarrow{W}\underset{\Delta_5=2}{\underline{\overrightarrow{W}\overleftarrow{\mathbf{D}}}}\overleftarrow{D}\overrightarrow{S}
\xrightarrow{\Delta_5}
\overleftarrow{W}\overrightarrow{W}\overrightarrow{W}\underset{\Delta_5=1}{\underline{H\overleftarrow{\mathbf{D}}}}\overrightarrow{S} \\
\xrightarrow{\Delta_2}
\overleftarrow{W}\overrightarrow{W}\overrightarrow{W}\underset{\Delta_5=0}{\underline{H\mathbf{H}}}\overrightarrow{S}
\xrightarrow{\Delta_2}
\overleftarrow{W}\overrightarrow{W}\overrightarrow{W}\underset{\Delta_5=0}{\underline{\mathbf{H}\overleftarrow{W}}}\overrightarrow{S}
\xrightarrow{\Delta_6}
\overleftarrow{W}\overrightarrow{W}\overrightarrow{W}\underset{\Delta_5=0}{\underline{\overrightarrow{\mathbf{W}}\overleftarrow{W}}}\overrightarrow{S}
\xrightarrow{\Delta_4}
\overleftarrow{W}\overrightarrow{W}\overrightarrow{\mathbf{W}}\underset{\Delta_5=3}{\underline{\overrightarrow{S}\overleftarrow{W}}}\overrightarrow{S} \\
\xrightarrow{\Delta_7}
\overleftarrow{W}\overrightarrow{W}\overrightarrow{S}\underset{\Delta_5=3}{\underline{\overrightarrow{\mathbf{S}}\overleftarrow{W}}}\overrightarrow{S}
\xrightarrow{\Delta_6}
\overleftarrow{W}\overrightarrow{W}\overrightarrow{S}\underset{\Delta_5=3}{\underline{\overrightarrow{\mathbf{D}}\overleftarrow{W}}}\overrightarrow{S}
\xrightarrow{\Delta_5}
\overleftarrow{W}\overrightarrow{\mathbf{W}}\underset{\Delta_5=2}{\underline{\overrightarrow{S}H}}\overleftarrow{W}\overrightarrow{S}
\xrightarrow{\Delta_2}
\overleftarrow{W}\overrightarrow{S}\underset{\Delta_5=2}{\underline{\overrightarrow{\mathbf{S}}H}}\overleftarrow{W}\overrightarrow{S} \\
\xrightarrow{\Delta_6}
\overleftarrow{W}\overrightarrow{S}\underset{\Delta_5=2}{\underline{\overrightarrow{D}\mathbf{H}}}\overleftarrow{W}\overrightarrow{S}
\xrightarrow{\Delta_6}
\overleftarrow{W}\overrightarrow{S}\underset{\Delta_5=2}{\underline{\overrightarrow{\mathbf{D}}\overleftarrow{W}}}\overleftarrow{W}\overrightarrow{S}
\xrightarrow{\Delta_5}
\overleftarrow{W}\underset{\Delta_5=1}{\underline{\overrightarrow{S}\mathbf{H}}}\overleftarrow{W}\overleftarrow{W}\overrightarrow{S}
\xrightarrow{\Delta_6}
\overleftarrow{W}\underset{\Delta_5=1}{\underline{\overrightarrow{S}\overleftarrow{W}}}\overleftarrow{W}\overleftarrow{W}\overrightarrow{S}
\end{array}
$$

Fig. 3. A computation: each configuration has a single anti-bond, which is underlined and labelled with the value of the Δ_5-distance; the bold letter is the letter to be changed; the arrow depicting each transition is labelled with the first decreasing Δ-component.

In all possible cases, distance Δ decreases, thus statement Prop of Theorem 1 holds. By Theorem 1, it follows that, whatever the starting configuration and the central schedule are, a configuration of $\mathcal{L}$ (with a single probabilistic token) will be reached within a finite time with probability 1. Note that, once a configuration of $\mathcal{L}$ has been reached, the subsequent configurations are all in $\mathcal{L}$ because $\mathcal{L}$ is closed under $\rightarrow$.

Appendix C: An Example of Finite Computation

Let us recall that, with our rewriting policy, once a bond is created, it never disappears (Δ_1 never increases) and always stays in fixed position. On the example below, there are two bonds, say B_1 and B_2, which correspond to the two leftmost and two rightmost letters of the configurations. Between B_1 and B_2 there lies an anti-bond A. Let us explain the general evolution of A between B_1 and B_2. At the beginning A is oriented rightwards and moves towards B_2 until it overlaps with it. It then looses its orientation and will become reoriented later in the other direction. Such a U-turn requires the rewriting of the left letter of B_2. Antibond A thus goes forth and back between B_1 and B_2. The rewriting of A is finite because every U-turn leads to the rewriting of the bond on which it "bounces", and a bond letter cannot be rewritten more than 3 times before becoming E. An iterated rewriting of A is illustrated in Figure 3. Each transition is represented by an arrow labelled with the first decreasing component of Δ. Δ_1 and Δ_3 always stay constant during the computation. The anti-bond of each configuration is underlined and labelled with the value of Δ_5: when A is oriented, Δ_5 corresponds to its π-distance; otherwise, Δ_5 is 0. In the last configuration, the anti-bond $\overrightarrow{S}\overleftarrow{W}$ cannot be rewritten without reaching $\mathcal{L}'$.