A Formal Framework for Clock-free Networks of Cells

Sergiu Ivanov

Institute of Mathematics and Computer Science
Academy of Sciences of Moldova
Academiei 5, Chişinău MD-2028 Moldova
sivanov@math.md
University of the Academy of Science of Moldova,
Faculty of Real Sciences,
3/2, Academiei str., MD-2028, Chişinău, Republic of Moldova

1 Introduction

Clock-free P systems were introduced in [7] as a natural extension to transitional membrane systems. The idea sparks from the observation of the fundamental difference between how transitional P systems evolve and how processes take place in biological cells: transitional P systems evolve in a series of crisp evolution steps, under the control of a global clock, while their biological prototypes have nothing similar to such a device.

In clock-free P systems, the attempt is made to bridge this difference by discarding any global step synchronisation mechanism. Any rule application lasts differently and there is no way of knowing when exactly the right-hand side of a rule will be added to the system.

The clock-free model produces two intuitive impressions. Firstly, it seems to be much closer to the real-world processes in the cell: the duration of clock-free rules is not regulated by any external mechanism and is expressed as a real number. Secondly, the absence of any built-in global step synchronisation seems to be very specific and quite unwieldy to manage. In fact, one of the best-working approaches to producing meaningful results with clock-free P systems is cutting down on parallelism as much as possible: this is how the computational completeness of these devices is shown in [7].

There are other ways to introduce time into P systems, an example could be timed P systems (see [1]). In this model, however, the global clock is still present.

In this paper we provide a formal description of the semantics of clock-free P systems, or rather the more general concept of clock-free networks of cells, and show how these devices can be modelled in transitional P systems.

This paper is heavily based on [3]. While we tried to introduce all the relevant concepts in this paper as well, getting acquainted with [3] would still be recommended.

2 Preliminaries

2.1 Multisets

Given a finite set A, by $|A|$ we understand the number of elements in A.

Let V be a finite alphabet; then V^* is the set of all finite strings of a V, and $V^+ = V^* - \{\lambda\}$, where λ is the empty string. By $\mathbb{N}$ we denote the set of all non-negative integers, by $\mathbb{N}^k$ – the set of all vectors of non-negative integers.

Let V be a finite set, $V = \{a_1, \ldots, a_k\}$, $k \in \mathbb{N}$. A *finite multiset* M over V is a mapping $M : V \to \mathbb{N}$. For each $a \in V$, $M(a)$ indicates the number of "occurrences" of a in M. The value $M(a)$ is called the *multiplicity* of a in M. The size of the multiset M is $|M| = \sum_{a \in V} M(a)$, i.e., the total count of the entries of the multiset. A multiset M over V can also be represented by any string x which contains exactly $M(a_i)$ instances of a_i, $1 \leq i \leq k$. The support of M is the set $\mathit{supp}(M) = \{a \in V \mid M(a) \geq 1\}$, which is the set which contains all elements of the multiset. For example, the multiset over $\{a, b, c\}$ defined by the mapping $\{a \to 3, b \to 1, c \to 0\}$ can be written as a^3b. The support of this multiset is $\{a, b\}$.

The class of all finite multisets over V is denoted by $\langle V, \mathbb{N} \rangle$. One may also consider mappings M of the form $M : V \to \mathbb{N}_\infty$, where $\mathbb{N}_\infty = \mathbb{N} \cup \{\infty\}$, i.e., the elements may have infinite multiplicity. A multiset M is infinite if $\big(\exists i \in \mathbb{N}\big)\big(1 \leq i \leq k\big)\big(M(a_i) = \infty\big)$, i.e., at least one element is of infinite multiplicity. The class of multisets M over V with $M : V \to \mathbb{N}_\infty$ is denoted by $\langle V, \mathbb{N}_\infty \rangle$. For $W \subseteq V$, W^∞ is the multiset in which every element is of infinite multiplicity: $\big(\forall a \in W\big)\big(W^\infty(a) = \infty\big)$.

Let $x, y \in \langle V, \mathbb{N}_\infty \rangle$ be two (possibly infinite) multisets over V. Then x is called a *submultiset* of y, written as $x \leq y$, if and only if $\big(\forall a \in V\big)\big(x(a) \leq y(a)\big)$. If $\big(\forall a \in V\big)\big(x(a) < y(a)\big)$ then x is called a strict submultiset of y. The sum of x and y, denoted by $x + y$ is defined in the following way: $\big(\forall a \in V\big)\big((x + y)(a) = x(a) + y(a)\big)$. The difference of x and y, denoted by $x - y$, is defined similarly: $\big(\forall a \in V\big)\big((x - y)(a) = x(a) - y(a)\big)$. The semantics of the symbol ∞ obey the usual rules: $\big(\forall n \in \mathbb{N}\big)\big(n \leq \infty \wedge \infty + n = \infty - n = \infty\big)$. When talking about $x - y$, we assume that $y \in \langle V, \mathbb{N} \rangle$, i.e., the that subtracted multiset is finite.

If $X = (x_1, \ldots, x_m)$ and $Y = (y_1, \ldots, y_m)$ are vectors of multisets over V, then the relation $X \leq Y$ is defined as follows $\big(X \leq Y\big) \Leftrightarrow \big(\forall i \in \mathbb{N}\big)\big(1 \leq i \leq m\big)\big(x_i \leq y_i\big)$, i.e., $X \leq Y$ if and only if each component of X is a submultiset of Y. Similarly, we define $X + Y$ and $X - Y$ in a component-wise way.

For further details on these topics see [2] and [6].

2.2 Clock-free P Systems

Clock-free P systems were originally introduced in [7]; an intuitive approach to working with the clock-free semantics was explored in [4].

A *clock-free* membrane system is defined by a tuple

$$\Pi = (O, C, \mu, w_1, w_2, \cdots, w_m, R_1, R_2, \ldots, R_m, i_0), \text{ where}$$

O	is a finite set of objects,
C	is a finite set of catalysts, $C \in O$,
μ	is a hierarchical structure of m membranes, bijectively labeled by $1, \ldots, m$; the interior of each membrane defines a region; the environment is referred to as region 0,
w_i	is the initial multiset in region $i, 1 \leq i \leq m$,
R_i	is the set of rules of region $i, 1 \leq i \leq m$,
i_0	is the output region.

The rules of a clock-free membrane system have the form $u \to v$, where $u \in O^+$ and $v \in (O \times Tar)^*$. In the case of non-cooperative rules, $u \in O$. The target indications from $Tar = \{here, out\} \cup \{in_j \mid 1 \leq j \leq m\}$ are written in the following way: (a, t), $a \in O$, $t \in Tar$ and the target *here* is typically omitted. A rule associated with membrane i must only specify a label of the immediately inner membrane in a target indication in_j.

The rules are applied in a maximally parallel way: no further rule should be applicable to the idle objects. In the case of non-cooperative systems, all objects evolve by the associated rules in the corresponding regions (except objects a in regions i such that R_i does not contain any rule $a \to u$, but these objects do not contribute to the result). Rules are non-deterministically chosen at each moment in time when a change occurs in the configuration of the P system. The process of choosing which rules should be applied does not take any time.

Intuitively, clock-free rule applications work in the following way. At the start of application, the multiset in the left-hand side of the rule is subtracted from the content of the corresponding region. When a rule application is complete, the multiset in the right-hand side of the rule is added to the corresponding region. The time between the start and the end of a rule application is a real value, may be different for different applications of the same rule, and there is impossible to know in advance.

For further definitions and details, see [7].

2.3 Networks of Cells

Networks of cells are a general framework for describing membrane systems with a static membrane structure. Networks of cells are formally described in depth in [3]. Intuitively, with this approach, membrane systems are considered as collections of interacting cells containing multisets of objects [8]. In this section we will only shortly expose the relevant considerations discussed in [3].

A *network of cells* of degree $n \geq 1$ is a tuple

$$\begin{aligned} \Pi &= (n, V, w, Inf, R), \text{ where} \\ n &\quad \text{is the number of cells,} \\ V &\quad \text{is a finite alphabet,} \\ w &= (w_1, \dots, w_n), w_i \in \langle V, \mathbb{N} \rangle, 1 \leq i \leq n, \text{is the initial content of cell } i, \\ Inf &= (Inf_1, \dots, Inf_n),\ Inf_i = \{a \in V \mid w_i(a) = \infty\}, 1 \leq i \leq n, \\ R &\quad \text{is a finite set of interaction rules.} \end{aligned}$$

According to the definition, the component Inf_i of the vector Inf contains the symbols which occur infinitely often in cell i. In most cases, only one cell, the environment, will contain symbols of infinite multiplicity.

The interaction rules in R have the form

$$(X \to Y, P, Q),$$

where $X = (x_1, \dots, x_n)$ and $Y = (y_1, \dots, y_n)$ are vectors of finite multisets over V, i.e., $x_i, y_i \in \langle V, \mathbb{N} \rangle$, $1 \leq i \leq n$. Furthermore, $P = (p_1, \dots, p_n)$ and $Q = (q_1, \dots, q_n)$, with $p_i, q_i \in \langle V, \mathbb{N} \rangle$, $1 \leq i \leq n$. The vector P is sometimes called the permitting condition of the rule, while Q is sometimes referred to as the forbidding condition of the rule.

The following notation for the rule $(X \to Y, P, Q)$ is also used:

$$\big((x_1, 1) \dots (x_n, n) \to (y_1, 1) \dots (y_n, n),\ (p_1, 1) \dots (p_n, n),\ (q_1, 1) \dots (q_n, n)\big).$$

Whenever any of x_i, y_i, p_i, or q_i, $1 \leq i \leq n$, is empty, it may be omitted.

Initially, every cell contains $w_i \cup Inf_i^\infty$. An interaction rule rewrites the objects x_i from cells i, $1 \leq i \leq n$, into objects y_j in cells j, $1 \leq j \leq n$, if $\big(\forall i \in \mathbb{N}\big)\big(p_i \leq x_i \wedge \neg(q_i \leq x_i)\big)$, i.e., if x_i contains p_i and does not contain q_i, $1 \leq i \leq n$.

Note that the definition of the network of cells does not specify any structural relations between the cells. The reason is that in many P system models the structural organisation of membranes is mainly used to direct communication between the cells (as can be seen in [5]). In networks of cells, however, rules are allowed to modify any combinations of cells, thus removing the need for an explicit structure of cells as a means of organising communication.

A *configuration* of the network of cells Π is an n-tuple of multisets over V: $C = (u'_1, \dots, u'_n)$, in which $u'_i \in \langle V, \mathbb{N}_\infty \rangle$, $1 \leq i \leq n$. Configurations are often described by their finite parts only: $C^f = (u_1, \dots, u_n)$, where $(u'_i = u_i \cup Inf_i^\infty) \wedge (u_i \cap Inf_i = \varnothing)$, $1 \leq i \leq n$ [3].

An interaction rule $r = (X \to Y, P, Q)$ is *eligible* in the configuration $C = (u_1, \dots, u_n)$ if and only if the following condition is true:

$$\Big(\forall i \in \mathbb{N}\Big)\Big(1 \leq i \leq n\Big)\Big((p_i \leq u_i) \wedge \big((q_i = \varnothing) \vee \neg(q_i \leq u_i)\big) \wedge (x_i \leq u_i)\Big),$$

i.e., the corresponding cells contain all of the promoting multisets, do not contain the forbidding multisets and contain the corresponding multisets of the left-hand side of the rule. The set of rules eligible in configuration C of the network of cells Π is denoted by $Eligible(\Pi, C)$.

Let $C = (u_1, \ldots, u_n)$ be a configuration of Π and C^f be its finite part. Let $T \in \langle R, \mathbb{N}\rangle$, $supp(T) \subseteq Eligible(\Pi, C)$, be a finite multiset of eligible rules, $|T| = k$. Recall that every eligible rule has the following form: $r = (X \to Y, P, Q) \in supp(T)$. The algorithm which checks whether the multiset of rules T can be applied to the configuration C is described in Algorithm 1. The algorithm checks if the rules in T can *all at once* be applied to the configuration C. To perform the check, an attempt is made to remove the left-hand sides of the rules in T from C. If this is possible, the algorithm returns the multiset union of the left-hand sides of the rules in T, otherwise it returns $\varnothing$. See [3] for further details.

Algorithm 1 The marking algorithm

```
1:  $T' \leftarrow T$
2:  $Mark_0(\Pi, C, T) \leftarrow (\lambda, \ldots, \lambda)$ {empty vector of size $n$}
3:  $i \leftarrow 1$
4:  while $T' \neq \varnothing$ do
5:      $r = (X \to Y, P, Q) \leftarrow$ get $T'$
6:      $T' \leftarrow T' - \{r \to 1\}$
7:      if $X \leq C^f - Mark_{i-1}(\Pi, C, T)$ then
8:          $Mark_i(\Pi, C, T) \leftarrow Mark_{i-1}(\Pi, C, T) + X$
9:      else
10:         return $\varnothing$
11:     end if
12:     $i \leftarrow i + 1$
13: end while
14: return $Mark_k(\Pi, C, T)$
```

If, for the multiset of eligible rules $T \in \langle R, \mathbb{N}\rangle$, $supp(T) \subseteq Eligible(\Pi, C)$, the marking algorithm succeeds and $Mark(\Pi, C, T) \neq \varnothing$, the multiset T is called applicable to C. The set of all multisets applicable to the configuration C of Π is denoted by $Appl(\Pi, C)$. The result of *applying* T to C is defined as follows:

$$Apply(\Pi, C, T) = C^f - \sum_{(X \to Y, P, Q) \in T} X + \sum_{(X \to Y, P, Q) \in T} Y,$$

or, equivalently,

$$Apply(\Pi, C, T) = (C^f - Mark(\Pi, C, T)) + \sum_{(X \to Y, P, Q) \in T} Y.$$

In plain words, $Apply(\Pi, C, T)$ is the configuration obtained from C by removing the left-hand sides of all rules in T and then adding the right-hand sides of all rules in T. Note that repeating entries of the same rule are treated independently both in $Mark$ and in *Apply*.

A *derivation mode* is a set of conditions applied to the set $Appl(\Pi, C)$ [3]. The maximally parallel derivation mode max is thus defined as follows:

$$Appl(\Pi, C, max) = \big\{T \in Appl(\Pi, C) \mid \big(\nexists T' \in Appl(\Pi, C)\big)\big(T' \supsetneq T\big)\big\}\,,$$

that is, $App(\Pi, C, max)$ contains those multisets of applicable rules which cannot be further maximised and remain applicable (which corresponds to the intuitive perception of the maximally parallel derivation mode).

Now fix a derivation mode ϑ (i.e. $\vartheta = max$). Consider two configurations C_1 and C_2 of Π. We say that $C_1 \Rightarrow_{(\Pi,\vartheta)} C_2$ if $\big(\exists T \in Appl(\Pi, C_1, \vartheta)\big)\big(C_2 = Apply(\Pi, C_1, T)\big)$, i.e., there exists an applicable multiset of rules T, valid under the derivation mode ϑ, with the property that applying T to C_1 yields C_2 [3]. In this case, C_2 is said to be the result of a transition step from C_1. When the network of cells and the derivation mode are clear from the context, the relation may be written as $\Rightarrow$. The reflexive and transitive closure of $\Rightarrow_{(\Pi,\vartheta)}$ is denoted by $\Rightarrow^*_{(\Pi,\vartheta)}$.

A configuration C of Π is said to be a halting configuration if C satisfies a certain halting condition. One of the most widely used halting conditions is $Appl(\Pi, C, \vartheta) = \varnothing$. Under this condition, C is a final configuration if there are no rules applicable to C under the derivation mode ϑ.

A *computation* of a network of cells Π under the derivation mode ϑ is the sequence of configurations $(C_i)_{i=0}^n$, where C_0 is the initial configuration, C_n is a halting configuration and $\big(\forall i \in \mathbb{N}\big)\big(0 \le i \le n-1\big)\big(C_i \Rightarrow_{(\Pi,\vartheta)} C_{i+1}\big)$. In plain words, a computation is a sequence of configurations which starts from the initial configuration and, by applications of multisets of rules valid under the derivation mode ϑ, reaches a halting configuration.

This section only contains a superficial overview of the corresponding material. Consider referring to [3] for further details on the formal framework for networks of cells.

3 Clock-free Networks of Cells

3.1 Preliminary Considerations

To understand and explore the concept of clock-freeness, we will start with an analysis of how a clock-free P system evolves. Consider the following sample clock-free P system:

$$\begin{aligned}
\Pi_0 &= (V, C, [\ \ ]_1, w_1, R_1, 1), \text{ where}\\
V &= \{a, b, c, x, u\},\\
C &= \{x\},\\
w_1 &= a^3,\\
R_1 &= \{a \to b, b \to c, a \to u, a \to x, xu \to x\}.
\end{aligned}$$

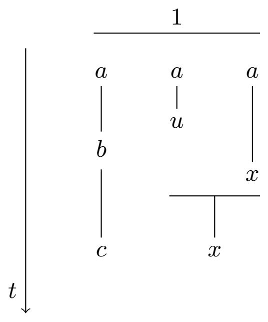

Fig. 1. The time diagram of a computation of Π_0

The time-diagram of a possible computation of Π_0 is shown in Figure 1. This diagram shows the possible evolution of individual symbols in the only region 1 of Π_0 as time progresses. In the initial state of the system, three different rules start consuming the three instances of a. At any moment sufficiently close to the initial state of the system, there are no symbols in region 1, because all of them have been consumed in the start of the three rule applications.

In this variant of evolution, the application of the rule $a \to u$ finishes first, producing an instance of u. Since there are no rules applicable to u, nothing happens at this time. The next rule application to finish is $a \to b$. At this moment, the contents of the only region of the system is bu. There is a rule $b \to c$, so, in accordance with the maximally parallel mode of evolution, this rule must be applied. The application of $b \to c$ therefore starts immediately after b has been produced and consumes the instance of b.

The next rule application to finalise is $a \to x$. As it can be seen from the diagram, at the moment when the first x is produced, the contents of region 1 will be ux. This renders the catalytic $xu \to x$ applicable, and so it is applied. In this variant of evolution, the applications of the rules $b \to c$ and $xu \to x$ finalise at exactly the same time. The system therefore halts with cx in its only region.

We explicitly remark that our choice of the variant of evolution is totally random. For example, the c may have not been produced at the same time with x. In fact, it could have been added to the system *before* u would be.

3.2 Clock-freeness: A Separate Concept

In this section we will tear apart the concept of a clock-free P system and focus on clock-freeness per se. The paper [7] introduces clock-free P system as a whole concept, without formally paying attention to the distinctive feature.

It turns out that clock-freeness cannot directly be described by any combination of the principal features of networks of cells considered in [3] (derivation mode, halting condition, goal of computations, interpretation of results). Indeed, the halting condition, the goal of computation, and the way to interpret results refer to the ending parts of the computation, while the derivation mode describes how to choose applicable multisets of rules. Clock-freeness, on the other hand, focuses on what happens after rules have been chosen, throughout the whole computation, not just in the closing phases.

The very special component of clock-freeness is that reaction times for rules are real numbers. This poses the question whether having real numbers in the model offers extended possibilities as compared to other models, where time is expressed as a natural number of steps. It turns out, however, that having arbitrary real numbers as reaction times is not at all defining. Indeed, the exact duration of a reaction is of no importance whatsoever. What deserves attention is only the relation of this value to other durations, i.e., we are only interested in knowing whether a multiset α was produced before, at the same time, of after multiset β. According to [7], we will consider the next configuration to be the instant description of the system when the output of a previously initiated reaction appears. Therefore, we consider the computation of a clock-free system to be a sequence of configurations, sampled at the moments when a rule application (or several rule applications) finalise. Observe that real numbers are relevant nowhere in this reasoning, since we always have a finite number of configurations in a computation.

Consider, for example, the computation shown in Figure 1. The sequence of configurations of this computation is the following:

$$(a^3), (u), (bu), (ux), (cx).$$

The computation starts with a^3 in the only region of the system. All three instances of a are consumed. Then a single u is produced. Later on, b is also added to the system. In this configuration, the rule $b \to c$ starts, so, when x is added to the system, there are no instances of b already. Finally, after both $b \to c$ and $ux \to x$ have finalised, the system stops in the halting configuration cx.

Observe again that we are not interested in reaction times themselves, but rather in the ordering of the moments of time when certain symbols were produced. Therefore, although the time intervals between configurations are some real numbers, we are free to discard this fact. Moreover, we can consider that a clock-free system transitions into the next configuration at every tick of a global clock, which brings us a huge step closer to the classic P system models. We lose absolutely nothing in this move because, as we have shown above, the exact duration of rules plays no role.

3.3 Formal Framework

Having done the preliminary consideration, we are now ready to introduce the clock-free mode of evolution into networks of cells. To be able to do that, we will extend the notion of a configuration.

Definition 1. *We call a* clock-free configuration *in a network of cells* $\Pi = (n, V, w, Inf, R)$ *the following construct:*

$$\begin{aligned} &C^* = (C, H),\ \textit{where} \\ &C = (u'_1, \dots, u'_n), u'_i \in \langle V, \mathbb{N}_\infty \rangle, 1 \le i \le n, \\ &H \in \langle R, \mathbb{N} \rangle. \end{aligned}$$

The initial *clock-free configuration of a network of cells is* $C^*_0 = (C_0, \varnothing)$*, where* C_0 *describes the initial content of every cell in the network.*

A configuration C^* has two components. C describes the contents of the cells of the system in exactly the same way as a normal configuration of a network of cells does. R is a multiset of rules which are "still being applied". The exact semantics of this component will be revealed in the next paragraphs.

The way a transition step from a configuration $C^*_1 = (C_1, H_1)$ into $C^*_2 = (C_2, H_2)$ under the derivation mode ϑ is performed is described in Algorithm 2. Symbolically, what Algorithm 2 does is build C^*_2 starting from C^*_1 such that $C^*_1 \Rightarrow_{(\Pi,\vartheta)} C^*_2$.

Algorithm 2 A clock-free transition step

```
1: $A \leftarrow$ **get** $Appl(\Pi, C_1, \vartheta)$
2: $C_2 \leftarrow C_1 - \sum_{(X \to Y, P, Q) \in A} X$
3: $H' \leftarrow H_1 + A$
4: $F \leftarrow$ **get submultiset** $H'$ {assure $F \neq \varnothing$}
5: $C_2 \leftarrow C' + \sum_{(X \to Y, P, Q) \in F} Y$
6: $H_2 \leftarrow H' - F$
```

The transition from C^*_1 starts by computing the set of multisets of applicable rules and picking one of them: A (line 1). The applications of the rules in A are started (line 2) and the rules themselves are added to the would-be second component of the new configuration (line 3). As remarked in the definition, the second component H_1 of the clock-free configuration $C^*_1 = (C_1, H_1)$ is a multiset of rules, whose applications have not yet been finalised. The algorithm continues with picking an arbitrary nonempty submultiset of rules F to be finalised from H' (line 4). The right-hand sides of the rules in F are added to the system (line 5), while the rules themselves are removed from H' (line 6).

Observe that if instead of choosing an arbitrary submultiset $F \le H'$, Algorithm 2 would take $F = H'$, it would degenerate into the classic (non-clock-free) algorithm of computing the next configuration from the current one.

Further note that the requirement $F \neq \varnothing$ does not limit the domain of configurations this algorithm is applicable to, because H' could only be empty if the algorithm started from a halting configuration.

Now that we have described a clock-free configuration and the way transitions between configurations occur, we are ready to formally define the halting condition for a clock-free computation.

Definition 2. *A clock-free configuration $C^* = (C, H)$ of a network of cells Π evolving under the evolution mode ϑ is* halting *when all rules have finalised and there are no more applicable rules:*

$$H = \varnothing \bigwedge Appl(\Pi, C, \vartheta) = \varnothing.$$

This halting condition corresponds to the clock-free semantics as described in [7]. Obviously, just as with other variants of networks of cells, the predicate defining the halting condition can be defined in a different way.

In what follows, we will refer to networks of cells with clock-free configurations, operating according to Algorith 2, as to *clock-free networks of cells.*

3.4 Example of Clock-free Evolution (Algorithm 2)

We will now turn back to the example of a computation of the clock-free P system Π_0 shown in Figure 1. The computation starts with the initial configuration:

$$C_0^* = \Big((a^3), \varnothing\Big).$$

In this configuration the multiset of rules to apply is selected to be $\{(a \to b) \to 1, (a \to u) \to 1, (a \to x) \to 1\}$; the three instances of a are correspondingly removed. The rule $(a \to u)$ is immediately picked to be finalised, so the next configuration is

$$C_1^* = \Big((u), \big\{(a \to b) \to 1, (a \to x) \to 1\big\}\Big).$$

Since there are no rules which consume only u, nothing is applicable in C_2^*, so $A = \varnothing$ once again. This time $F = \{(a \to b) \to 1\}$, so the system transitions into

$$C_2^* = \Big((bu), \big\{(a \to x) \to 1\big\}\Big).$$

In this configuration the rule $b \to c$ becomes applicable, so its application must be started: the only b is removed and H' is correspondingly modified. The rule $a \to x$ is finalised ($F = \{(a \to x) \to 1\}$):

$$C_3^* = \Big((ux), \big\{(b \to c) \to 1\big\}\Big).$$

There is the rule $xu \to x$, so it must be started in this configuration. The algorithm then picks both rules which are "still being applied" (including the $xu \to x$, which has just been started) and finalises them:

$$C_4^* = \Big((cx), \varnothing\Big).$$

Since no rules are applicable and $H_4 = \varnothing$, the system has arrived at a halting configuration.

3.5 Suitability of the Formalisation

In this section we will discuss whether the formalism introduced and described in the previous section is compatible with the intuitive description of clock-freeness provided in [7]. In this paper we define clock-freeness on the foundation of networks of cells, while [7] starts from transitional P systems. To bridge the obvious gap, we will consider networks of cells with rules working in clock-free mode. This will bring a common ground to the (extended) definitions from [7] and the formal definitions suggested in this paper.

Definition 3. *A* *-network of cells *is a network of cells with rules operating in clock-free mode, in the sense of [7].*

According to the definition, rule applications in a *-network of cells last for a different real-valued time interval each. In this section we will only consider $\vartheta = max$ for both *-networks of cells and clock-free networks of cells, because clock-free P systems evolve under maximal derivation mode. The halting condition for *-networks of cells will be the condition that all rule applications have finalised and no more rules are applicable, while networks of cells with clock-free configurations will have the halting condition introduced in the previous sections. Since the goal of the computation and the way to interpret the result do not directly pertain to the subject of this section, we will consider these two parameters as having a certain well-defined value, the same for both kinds of analysed networks of cells.

Obviously, clock-free P systems as defined in [7] are a particular case of *-networks of cells.

Theorem 1 (Suitability of the formalism). *Consider a network of cells $\Pi = (n, V, w, Inf, R)$ and a finite sequence of clock-free configurations of Π: $\mathcal{C}^* = (C_i^*)_{i=0}^m$, with $C_0^* = (C_0, \varnothing)$ being an initial configuration and $C_m^* = (C_m, \varnothing)$ being a halting configuration. Let ${}^*\Pi = (n, V, w, Inf, R)$ be a *-network of cells. $\mathcal{C}^*$ is a clock-free computation if and only if the sequence of the first components $\mathcal{C} = (C_i)_{i=0}^m$ is a valid computation in ${}^*\Pi$.*

Proof. According to the corresponding definitions, C_0^* and C_n^* are valid clock-free initial and halting configurations in Π correspondingly if and only if C_0 and C_n are valid initial and halting configurations in ${}^*\Pi$ correspondingly. Obviously, the statement of the theorem holds for $C_0^* = C_m^*$, therefore we will focus on the cases when $m > 0$.

Consider C_0^* and C_1^* and suppose that $C_0^* \Rightarrow_{(\Pi,max)} C_1^*$. Then, in ${}^*\Pi$, we can consider the transition from C_0 to C_1 constructed in the following way: start the applications of the rules belonging the multiset A chosen in Algorithm 2, then consider that the rules collected into F as constructed in Algorithm 2 finalise their application at one and the same time. The moment these rules complete is the moment when C_1 will occur. Therefore, $C_0 \Rightarrow_{({}^*\Pi,max)} C_1$.

Now suppose that $C_0 \Rightarrow_{({}^*\Pi,max)} C_1$. This means that, in ${}^*\Pi$, some rule applications started in C_0, and some of these rules finished to result in configuration

C_1. We can therefore consider that in configuration C_0^*, Algorithm 2 chose to start the same rules as the ones which started in in ${}^*\Pi$ and then immediately finalised those which led to the occurrence of C_1 in ${}^*\Pi$. For C_1^* constructed in this way, the following is true: $C_0^* \Rightarrow_{(\Pi,max)} C_1^*$.

We have therefore proved that

$$\big(C_0^* \Rightarrow_{(\Pi,max)} C_1^*\big) \Leftrightarrow \big(C_0 \Rightarrow_{({}^*\Pi,max)} C_1\big).$$

Moreover, we have proved that H_1 contains those and only those rules which could have been started in ${}^*\Pi$ in configuration C_0 and might have not been finalised in transition to C_1. By repeating the same reasoning for any pair C_i^* and C_{i+1}^*, it is now possible to prove the statement of the theorem by induction.

4 Simulations of Clock-freeness

Now that we have formally defined clock-freeness, it is time to pose one of the most important questions: how "far away" are clock-free networks of cells from the traditional networks of cells? It turns out that it is very easy to simulate clock-freeness with traditional, static networks of cells, as they are formalised in [3].

Indeed, consider a clock-free network of cells $\Pi = (n, V, w, Inf, R)$ and a rule $r_i = (X \to Y, P, Q) \in R'$. According to clock-free semantics, an application of r_i is started in a configuration C_i^* of Π by removing its left-hand side from the system, and is finalised in a configuration C_j^* by adding its right-hand side to the system. Consider now an ordinary network of cells $\Pi' = (n, V', w, Inf, R')$, with R' and V' constructed in the following way:

$$\begin{aligned}
V' &= V \cup \big\{\xi_i \,\big|\, r_i = (X \to Y, P, Q) \in R\big\},\\
R' &= \Big\{\big(X \to (\xi_i, k^*), P, Q\big), \big(((\xi_i, k^*) \to (\xi_i, k^*), \bar\lambda, \bar\lambda\big),\\
&\qquad \big((\xi_i, k^*) \to Y, \bar\lambda, \bar\lambda\big) \,\big|\, r_i = (X \to Y, P, Q) \in R\big\},\\
&\qquad 1 \le k^* \le |R|,
\end{aligned}$$

where $\bar\lambda$ is the vector of size n of empty multisets. The alphabet of Π' includes all symbols from the alphabet of Π, but also a ξ_i per each rule with index i.

For each rule in R, three rules are added to R'. When the rule r_i is applicable, instead of X being directly transformed into Y, X is initially rewritten into ξ_i, which is placed into the cell with index k^*. The choice of the index k^* is totally arbitrary, it may even be different for different rules. The symbol ξ_i can either reproduce itself or add Y to the system; either of these will unconditionally happen; the choice between the two options is nondeterministic.

We claim that Π' accurately simulates the evolution of Π. Indeed, a rule $(X \to (\xi_i q, k^*), P, Q) \in R'$ is applicable in Π' if and only if the corresponding rule $r_i = (X \to Y, P, Q) \in R$ is applicable. Rewriting X into ξ_i thus corresponds to removing the left-hand side of the rule from the system and adding it to H' (the operations performed in Algorithm 2). If, in a certain configuration, ξ_i is rewritten into itself, then, at this step, the rule r_i was not picked by Algorithm 2 to be finalised. The case when ξ_i is rewritten into Y corresponds to the scenario when Algorithm 2 picked rule r_i to finalise.

Observe now that if, in a certain configuration of Π', only the rules which rewrite ξ_i, $1 \le i \le |R|$, were chosen to be applied, then the system will arrive at exactly the same configuration at the next step. It is possible to detect such situations and cut off such computations. This however, does not make the following result significantly different on the overall.

Theorem 2 (Simulation of clock-freeness). *Consider a clock-free network of cells $\Pi = (n, V, w, Inf, R)$ and a sequence of clock-free configurations $\mathcal{C}^* = (C_i^*)_{i=0}^m$, with C_0^* being an initial configuration and C_m^* being a halting configuration. If $\mathcal{C}^*$ is a computation then it is possible to construct a computation $\mathcal{K} = (K_i)_{i=0}^l$ of an ordinary network of cells Π' so that there is a mapping $f : \mathcal{K} \to \mathcal{C}^*$ with the properties:*

1. *$f(K_i) = C_j^* = (C_j, H_j) \Leftrightarrow C_i \le K_i$ ($f(K_i) = C_i^*$ if and only if K_i contains at least all the symbols in K_i),*
2. *$\big(K_i \Rightarrow^*_{(\Pi',max)} K_j\big) \Leftrightarrow \big(f(K_i) \Rightarrow^*_{(\Pi,max)} f(K_j)\big)$ (f maps the binary relation $\Rightarrow^*_{(\Pi',max)}$ into $\Rightarrow^*_{(\Pi,max)}$ and vice versa),*
3. *$\big(\forall i\big)\big(\exists j\big)\big(f(K_j) = C_i^*\big)$ (f is surjective),*

where $1 \le i \le m$, $1 \le j \le l$.

Proof. According to the constructions in the previous paragraphs, to a clock-free network of cells Π, one can associate an ordinary network of cells Π' which simulates Π. The mapping f can thus be defined as follows: from the region i^* of K_i remove all instances of ξ_i, $1 \le i \le |R|$; this will be the first component of $f(K_i)$. The second component of $f(K_i)$ is obtained by starting with an empty multiset and adding an instance of rule r_i per each instance of symbol ξ_i, $1 \le i \le |R|$. The required properties of f follow from the constructions shown in this section.

5 Conclusion

In this paper we have torn apart the concept of clock-free P systems as defined in [7] and have separated clock-freeness as a stand-alone ingredient. We have formally defined this ingredient within the framework of networks of cells [3] and shown that this definition is consistent with the original concept. We have also shown that clock-freeness can be simulated with usual networks of cells in a quite straightforward way.

The fact that clock-freeness can be simulated so easily goes against the intuitive impression produced by clock-free P systems and poses the important question of how valuable this ingredient is. Indeed, it seems that almost any problem in clock-free systems can be equivalently formulated for the corresponding clocked systems.

We believe that clock-freeness is fairly important, though, because it is (intuitively) much closer to how processes take place in biological cells. The fact that clock-freeness is easy to simulate is thus beneficial and shows how it is possible to move closer to real life without sacrificing too much.

We remark, of course, that the chemical reactions taking place in the cell have been studied well enough to approximately predict their durations or, at least, compare them to other cellular processes in terms of speed. Clock-freeness, however, allows us to abstract away these details. Metaphorically put, an implementation of an operation in a clock-free network of cells (or clock-free P system) can survive changes in the physical implementation, because it does not depend on the durations of underlying chemical relations. This reasoning is of course hypothetical at the moment, but it may become practical quite soon.

In this paper we have used the term "clock-freeness" to denote the mode of evolution of a network of cells in which rule applications may last for an arbitrary long or short amount of time. The word "free" in the term "clock-free", therefore, refers to a different concept than the same word in the term "time-free" [1]. However, the majority of clock-free P systems considered in [7] and, for example, [4], are in fact independent of the what the durations of rules are. It thus is possible to consider that the terms "clock-freeness" and "time-freeness" do have something in common. Observe that Theorem 2 allows translating the problem of independence of the durations of rules in clock-free networks of cells into the problem of confluence in regular networks of cells.

Acknowledgements

The author gratefully acknowledges Artiom Alhazov, PhD for the remarkably fruitful discussions during the work on [4], without which the present paper would not have happened in the first place.

References

1. A. Alhazov, M. Cavaliere: Evolution-communication P Systems: Time-Freeness. *Proc. Third Brainstorming Week on Membrane Computing* (M.A. Gutiérrez-Naranjo et al. eds.), 2005, 11–18.
2. J. Dassow, Gh. Păun: On the power of membrane computing. *Journal of Universal Computer Science*, 5 (1999), 33–49.
3. R. Freund, S. Verlan: A Formal Framework for Static (Tissue) P Systems. *8th International Workshop on Membrane Computing, WMC2007* (G. Eleftherakis et al., eds.), LNCS 4860, Springer, 2007.
4. S. Ivanov: Basic Concurrency Resolution in Clock-Free P Systems. *Proc. 12th International Conference on Membrane Computing* (M. Gheorghe et al., eds.), LNCS 7184, Springer, 2012, 226–242.
5. Gh. Păun: *Membrane Computing. An Introduction.* Springer, Berlin, 2002.
6. G. Rozenberg, A. Salomaa, eds.: *Handbook of Formal Languages*, 3 vols., Springer, Berlin, 1997.

7. D. Sburlan: Clock-free P Systems. *Pre-proc. Fifth Workshop in Membrane Computing* (G. Mauri, Gh. Paun, C. Zandron, eds.), 2004, 372–383.
8. F. Bernardini, M. Gheorghe, M. Margenstern, S. Verlan: Networks of Cells and Petri Nets. *Proc. 5th Brainstorming Week on Membrane Computing* (M.A. Gutiérrez-Naranjo et al., eds.), 3 2007, 3–62.
9. The P Systems Web Page: `http://ppage.psystems.eu`.