Synchronized Rules in Membrane Computing

Péter Battyányi

Department of Computer Science, Faculty of Informatics
University of Debrecen

2024 January

Motivation

Synchronization of rules were defined by Aman and Ciobanu.[1] Assume we have a symbol-object P system comprising one cell with configuration a^3 and with maximally parallel execution mode. Let

$$r_1 : a \to b$$
$$r_2 : a \to c.$$

We consider $r = r_1 \otimes r_2$. Then r can be applied only if both r_1 and r_2 can simultaneously be applied and, in this case, the multiplicities of r_1 and r_2 are determined by the maximally parallel rule. Hence, we either have

$$a^3 \to^r b^2 c,$$

or

$$a^3 \to^r bc^2.$$

---

[1] Aman, B., Ciobanu, G. Synchronization of rules in membrane computing. J. Membr. Comput. 1, 233-240 (2019)

Motivation

Another example[2] is a P system of order one with configuration a^2b^3u such that

$$\begin{aligned} r_1 &: \quad b \to bd \\ r_2 &: \quad au \to u. \end{aligned}$$

Let $\rho = \{r_1 \otimes r_2\}$. Then

$$\begin{aligned} a^2b^3u &\to ab^3d^3u \\ &\to b^3d^6u. \end{aligned}$$

[2]Aman, B., Ciobanu, G. Synchronization of rules in membrane computing. J. Membr. Comput. 1, 233-240 (2019)

Definition

- The tuple $\Pi = (O, w_1^0, w_2^0, \ldots, w_n^0, (R, \rho), i_o)$ is called a *P* system with synchronization of rules (or with synchronized rules) of degree $n \geq 1$ if $\Pi' = (O, w_1^0, w_2^0, \ldots, w_n^0, R, i_o)$ is a symbol-object P system of degree n and $\rho \subseteq \cup_{j=1}^{n} R_j^{\geq 2}$.
- Let $1 \leq j \leq n$ be given. For any element $(r_1, r_2, \ldots, r_k)$ of ρ, where $r_i \in R_j$ $(1 \leq i \leq k)$, we use the notation $r = r_1 \otimes r_2 \otimes \ldots \otimes r_k$ and we call r_i the components $(1 \leq i \leq k)$. By abuse of notation, we write $r \in R_j$. We term r a single rule if it is not a synchronized one. We may assume that no rule $r \in R$ is a single rule and a component of a synchronized rule at the same time.

Synchronized rules

Let $\Pi = (O, w_1, w_2, \ldots, w_n, (R, \rho), i_o)$ be a P system with rule synchronization. Let $r \in R_j$ for some $1 \leq j \leq n$. If r is a single rule, its application in Π is defined as originally in P systems. For a synchronized rule $r = r_1 \otimes r_2 \otimes \ldots \otimes r_k$, we distinguish four application modes.

A1 r is applicable when all the r_i's are applicable as single rules ($1 \leq i \leq k$). Assuming the applicability of r, we apply r by executing each component of r at least once. The newly created objects do not take part in the actual rule application step.

A2 r is applicable when all the r_i's are applicable as single rules ($1 \leq i \leq k$). Assuming the applicability of r, we apply r by executing each component of r at least once. The execution of the components of r is sequential. The newly created objects can be reused in the actual rule application step.

Synchronized rules

B1 r is applicable when at least one of r_i's is applicable as a single rule ($1 \leq i \leq k$). Assuming the applicability of r, we apply r by executing the applicable component of r in the order appearing in r. The newly created objects do not take part in the actual rule application step.

B2 r is applicable when at least one of r_i's is applicable as a single rule ($1 \leq i \leq k$). Assuming the applicability of r, we apply r by executing the applicable component of r in the order appearing in r. The newly created objects can be reused in the actual rule application step.

Synchronized rules

The application modes starting with letter A are called weak application modes, while the ones starting with B are the strong application modes. In what follows, we give a short account of the computational strength of rule synchronization with the various execution modes. We examine the case B2 in more detail. The other cases are still subject to investigations. Since rule synchronization behaves smoothly with flattening, in the sequel we assume that our P system Π is of degree 1.

Case of B2

We turn to the case of execution mode B2. This means that we apply the strong mode of execution together with the possibility of reusing the newly created objects during a rule application. We prove that an arbitrary context-free matrix grammar with appearance checking can be simulated by a P system with synchronized rules in B2 execution mode.

Case of B2

Let $G = (N, T, S, M, F)$ be a context-free matrix grammar in the binary normal form. That is, there exist the following three sets that are mutually disjoint such that $N = N_1 \cup N_2 \cup \{S, \#\}$ and the matrices are as below:

1. $(S \rightarrow X_{ini} A_{ini})$, with $X_{ini} \in N_1$, $A_{ini} \in N_2$,
2. $(X \rightarrow Y, A \rightarrow x)$, with $X, Y \in N_1$, $A \in N_2$, $x \in (N_2 \cup T)^*$, $|x| \leq 2$,
3. $(X \rightarrow Y, A \rightarrow \#)$, with $X, Y \in N_1$, $A \in N_2$,
4. $X \rightarrow \lambda, A \rightarrow x)$, with $X \in N_1$, $A \in N_2$, $x \in T^*$, $|x| \leq 2$,

where the rules $A \rightarrow \#$ comprise F, and $\#$ is a trap-object. Furthermore, there is only one matrix of type 1, and a matrix of type 4 is used only once, in the end of the derivation.

We construct a P system $\Pi = (O, [_1]_1, HX_{ini}A_{ini}, R_1, 1)$ of degree one with alphabet $O = N \cup \{H, H'\}$ and with the following set of rules R_1:

1. $HXA \to HYx$ for the matrices of type 2,
2. $r_1^M : HX \to H'Y$, $r_2^M : H'A \to \#$, $r_3^M : H' \to H$ for the matrices M of type 3,
3. $HXA \to Hx$ for the matrices of type 4.

Moreover, we let $\rho = \{r_1^M \otimes r_2^M \otimes r_3^M \mid M \text{ is of type } 3\}$.

Case of B2

One may have observed that the rules applied in the P system with rule synchronization were cooperative ones. The question arises whether this is an inherent property of the system or we could remove the cooperativeness property somehow. We show that non-cooperative rules do not provide the same computational strength. Namely, we prove that there exists no P system of degree one with non-cooperative synchronized rules and execution mode B2 which computes the function

$$\overline{sg}(n) = \begin{cases} 1 & \text{if } n = 0, \\ 0 & \text{otherwise.} \end{cases}$$

Case of B2

Let $\Pi = (O, w_0, (R_1, \rho), 1)$ be a P system with synchronized non-cooperative rules of degree one computing the $\overline{sg}$ function. We make some natural stipulations. First of all, let the natural number n as input be represented by the multiset a_0^{n+1}, and suppose that the output m is represented by the multiset a_1^{m+1} provided the computation halts, where $O = \{a_0, a_1, \ldots, a_k\}$. We simply write $\overline{n}$ for input value n and $\underline{m}$ for output value m. We say that a configuration c is in normal form if no rule can be applied to it. In notation: $c \in NF$. Otherwise, there exist c' and r such that $c \Rightarrow^r c'$. We make the stipulation that, for every input $\overline{n}$, a computation of Π either does not halt or, when it halts, it provides the result $\overline{sg}(n)$. This implies that, for the configurations $\underline{0}$ and $\underline{1}$, we have $\underline{0}, \underline{1} \in NF$. Moreover, if c is a configuration of Π, we use the shorthand $c(i)$ for $c(a(i))$ $(1 \leq i \leq k)$ provided $O = \{a_1, \ldots, a_k\}$.

Let c, c' be two configurations. We denote $c \preceq c'$ if, for every $1 \leq i \leq k$, $c(i) \leq c'(i)$.

Proposition

Let c, c' be such that $c \preceq c'$. Then $c' \in NF$ implies $c \in NF$.

Observe that, for any $r \in R_1 \cup \rho$, if r is applicable for c', then r is applicable for c.

Case of B2

Lemma

Let $\bar{c}$, c be such that $\bar{c} \preceq c$. Assume $c \Rightarrow^{\sigma} c'$ for some c' and σ. Then $\exists\ \bar{\sigma}$ and $\bar{c}'$ such that $\bar{c} \Rightarrow^{\bar{\sigma}} \bar{c}'$ and $\bar{c}' \preceq c'$.

Proof

Let $\bar{c}$, c, c' and σ as above. Assume $|\sigma| = n$. We argue by induction on n.

- $n = 0$. Trivial.
- $n = m + 1$. Let $\sigma = \sigma''\#[r]$. Assume $c \Rightarrow^{\sigma''} c''$. Then $\exists\ \bar{\sigma}''$ and $\bar{c}''$ such that $\bar{c} \Rightarrow^{\bar{\sigma}''} \bar{c}''$ and $\bar{c}'' \preceq c''$.

Proof cont

- r is a single rule. If $dom(r) \in \bar{c}''$, then let $\bar{\sigma} = \bar{\sigma}''\#[r]$. Then the statement is obvious. Otherwise, let $\bar{\sigma} = \bar{\sigma}''$. In this case, $a = dom(r) \notin \bar{c}''$ implies $\bar{c}''(a) = \bar{c}'(a) = 0$. For the other components of the multisets the kineqaulityis obviously preserved.
- $r = r_1' \otimes \ldots \otimes r_l'$. We argue by the number of intermediate reductions by which r is applied to c'' to obtain c'.

Proof cont

Let $c'' \to^{\tau} c'''$. Let us write $|\tau| = p$.

* $p = 0$. Obvious.
* $p = q + 1$. Then we have $c'' \to^{\tau'} c'''' \to^{[r']} c'''$ for some $\tau = \tau'\#[r']$ and r' is a component of r, say r_i, for which $\exists$ $'\bar{c}'' \to^{\bar{\tau}'} \bar{c}''''$ such that $\bar{c}'''' \preceq c''''$. If $dom(r') \in \bar{c}''''$, then let the next component of r to be applied for $\bar{c}''''$ is r', since, otherwise $dom(r'') \in \bar{c}''''$ and $r'' = r_j$ with $j < i$, together with $\bar{c}'''' \preceq c''''$ would imply that r'' should have been executed for c'''' instead of r'. If $dom(r') \notin \bar{c}''''$, then let $\bar{c}''' = \bar{c}''''$.

Case of B2

Theorem

Let Π be a P system of order one with rule synchronization and with non-cooperative rules. Then Π is not computationally complete.

Proof

We show that Π cannot compute the $\overline{sg}$ function. Let $\bar{0}$ and $\bar{1}$ as above. Then $\bar{0} \preceq \bar{1}$. Let $\bar{1} \Rightarrow^* \underline{1}$ be a halting computation for $\bar{1}$. By the previous lemma, there exists a configuration c such that $\bar{0} \Rightarrow^* c$ and $c \preceq \underline{1}$. Since $\underline{1} \in NF$, we have $c \in NF$. But Π computes the $\overline{(sg)}$ function, hence $c = \underline{0}$. A contradiction.

Thank you for your attention!