P Systems with Chained Rules

Dragoş Sburlan
dsburlan@univ-ovidius.ro
Ovidius University of Constanta

CMC12, Fontainebleau, France
23-26 August 2011

Motivations

A necessary condition (but not sufficient) in the principle of causality is the delay of one event relatively to another. The sufficient condition imposes the precise definition of what event is the cause and what event is the effect.

The principle of causality exhibits a certain temporal order by which any later event is determined by the earlier one.

Motivations

"Bio-chemical reactions take place at the level of cell compartments not only by creating, transforming, eliminating, transporting substances, but also by acting on the compartments themselves (creating and/or destroying compartments)."

Traditionally, while modeling such systems, the aim is to capture the execution of each process that might occur. One common feature of the proposed formal models was the fulfilling of the principle of causality: every event has a cause (for instance, the execution of some bio-chemical reaction triggers the execution of others). This principle was "implemented" in the formal models by specifying that the resulted compounds (produced by the execution of a rule), will trigger the execution of other rule(s).

Prerequisetes

- Lindemayer systems (in particular ET0L systems) and the corresponding families of length sets of languages ($NET0L$);

- register machines and recursively enumerable sets of natural numbers (NRE).

It is known that ET0L systems using two tables generate the same family of languages as the one generated by arbitrary ET0L systems.

It is known that $NCF \subset NET0L \subset NRE$. For instance the set $\{2^n \mid n \geq 0\} \in NET0L \setminus NCF$.

P Systems with Chained Rules I

A *P system with chained rules* (a PCR system, for short) of degree $m \geq 1$ is a construct

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

- O is an alphabet of *objects*;
- $C \subseteq O$ is the set of *catalysts*;
- μ is a tree structure of $m \geq 1$ uniquely labelled *membranes*; usually, the set of labels is $\{1, \ldots, m\}$;
- $w_i \in O^*$, for $1 \leq i \leq m$, are multisets of objects which are initially present in the regions of μ (as indicated by the index);
- R_i, $1 \leq i \leq m$, are finite sets of *vectors of evolution rules*. A vector of evolution rules is denoted as $(r_1, \ldots, r_k)$ and each r_i, $1 \leq i \leq k$, is a *non-cooperative* rule $a \to v$ or a *catalytic* rule $ca \to cv$, where $a \in O \setminus C$, $v \in ((O \setminus C) \times \{here, out, in\})^*$, and $c \in C$;
- $i_0 \in \{1, \ldots, m\}$ is the label of the *output region* of Π.

P Systems with Chained Rules II

A *configuration* of Π is a vector $(\alpha_1, \ldots, \alpha_m)$, where $\alpha_i \in O^*$, $1 \leq i \leq m$, is the multiset of objects present in the region i of Π. The *initial configuration* of Π is the vector $C_0 = (w_1, \ldots, w_m)$.

Let $R_i = \{v_{(i,1)}, \ldots, v_{(i,s_i)}\}$ where $1 \leq i \leq m$ and such that $s_i \geq 1$; in addition, let $v_{(i,j)} = (r_{(i,j,1)}, r_{(i,j,2)}, \ldots, r_{(i,j,t_j)})$, $1 \leq j \leq s_i$ and such that $t_j \geq 1$. In other words, the first index of a rule indicates the region where it belongs, the second index indicates the vector from which it belongs, and finally the last index indicates the position in the vector.

P Systems with Chained Rules III

A computation of Π is a recursively defined sequence of configurations (possibly infinite)

$$C_0, C_1, \ldots, C_k, C_{k+1}, \cdots$$

having the following properties:

- C_0 is the initial configuration;
- $C_{k+1} = (w_{(k+1,1)}, \ldots, w_{(k+1,m)})$ is obtained from $C_k = (w_{(k,1)}, \ldots, w_{(k,m)})$ by applying in a nondeterministic manner on each multiset $w_{(k,i)}$, $1 \leq i \leq m$, a maximal multiset of rules (with competition on objects) of the following form (the multiset is expressed as a string indicating a union of multisets of rules belonging to $v_{(i,j)}$, $1 \leq j \leq s_i$; also, the multiplicities of some rules might be 0):

$$\overbrace{r_{(i,1,1)}^{p_{(i,1,1)}} r_{(i,1,2)}^{p_{(i,1,2)}} \cdots r_{(i,1,t_1)}^{p_{(i,1,t_1)}}}^{\text{a multiset of rules from } v_{(i,1)}} \quad \cdots\cdots \quad \overbrace{r_{(i,s_i,1)}^{p_{(i,s_i,1)}} r_{(i,s_i,2)}^{p_{(i,s_i,2)}} \cdots r_{(i,s_i,t_{s_i})}^{p_{(i,s_i,t_{s_i})}}}^{\text{a multiset of rules from } v_{(i,s_i)}} .$$

P Systems with Chained Rules IV

In addition, this multiset of rules has to obey the following conditions: if $p_{(i,s,t)} \geq 1$, $1 \leq s \leq s_i$, $1 \leq t \leq t_{s_i}$ (i.e., the corresponding rule $r_{(i,s,t)}$ will be applied) then it is obligatory that the rule $r_{(i,s,t-f)}$ for $1 \leq f \leq t-1$ was applied to the configuration C_{k-f} (providing that $k - f \geq 0$).

P Systems with Chained Rules V

In other words, given the multiset $w_{(k,i)}$ of the configuration C_k, if a vector of chained rules v starts its *application* (that is, the first rule in the vector is applied), then in the next subsequent configurations the rest of the rules from v will be applied in order in consecutive steps(one rule by each subsequent configuration but such that every such rule can be applied several times). However, if a rule from a given position f in an already started vector of chained rules v cannot be applied (although it was supposed to be executed *), then the execution of v is dropped (that is, for the current application of v, the remaining rules are not executed anymore).

*This happens when either in the current multiset there is no object that triggers the execution of the rule, or it was a competition on the objects and this rule lost it.

An example

The following P system with chained rules

$$\Pi = (O, C, \mu, w_1, R_1, i_0)$$

where

$$\begin{aligned} O &= \{a, b\}\ ; \\ C &= \emptyset\ ; \\ \mu &= [\,]_1\ ; \\ w_1 &= ba\ ; \\ R_1 &= \{v_{(1,1)} : (b \to b, a \to aa), v_{(1,2)} : (b \to \lambda)\}\ ; \\ i_0 &= 1. \end{aligned}$$

generates the set $\{2^n \mid n \geq 0\}$ (the well known non-semilinear set of natural numbers).

Step	**Current Config.**	**Applied vectors/rules**	**Next Config.**
0	$C_0 = ([\,]_1, ba)$	Nondeterministic choice 1: $(\boldsymbol{b \to b}, a \to aa)$	C_1^1
		Nondeterministic choice 2: $(\boldsymbol{b \to \lambda})$	C_1^2
1	$C_1^1 = ([\,]_1, ba)$	Nondeterministic choice 1: $(b \to b,\ \boldsymbol{a \to aa})$ $(\boldsymbol{b \to b}, a \to aa)$	C_2^1
		Nondeterministic choice 2: $(b \to b,\ \boldsymbol{a \to aa})$ $(\boldsymbol{b \to \lambda})$	C_2^2
	$C_1^2 = ([\,]_1, a^{2^0}) = ([\,]_1, a)$	-	-
⋮			
k	$C_k^1 = ([\,]_1, ba^{2^{k-1}})$	Nondeterministic choice 1: $(b \to b,\ \boldsymbol{a \to aa})$ $(\boldsymbol{b \to b}, a \to aa)$	C_{k+1}^1
		Nondeterministic choice 2: $(b \to b,\ \boldsymbol{a \to aa})$ $(\boldsymbol{b \to \lambda})$	C_{k+1}^2
	$C_k^2 = ([\,]_1, a^{2^{k-1}})$	-	-
⋮			

Results I

P systems with chained rules using vectors composed only by non-cooperative rules can generate at least the same family of sets of numbers as $NET0L$.

$$NOPCRP_1(ncoo) \supseteq NET0L.$$

$$\begin{aligned}
R_1 \;=\;& \{(t \to t_{(1,0)}^{card(T_1)} X), (t \to t_{(2,0)}^{card(T_2)} X)\} \\
\cup\;& \{(t_{(i,0)} \to t_{(i,1)}, \overline{r_{(i,j)}}) \mid i \in \{1,2\}, 1 \le j \le s_i\} \\
\cup\;& \{(t_{(1,1)} \to t_{(1,2)}), (t_{(2,1)} \to t_{(2,2)})\} \\
\cup\;& \{(t_{(i,j)} \to t_{(i,j+1)}, a_{(j-1)} \to \#) \mid i \in \{1,2\}, 2 \le j \le n+1\} \\
\cup\;& \{(\# \to \#)\} \\
\cup\;& \{(t_{(i,j)} \to t_{(i,j+1)}, h(a_{j-n-1}) \to a_{j-n-1}) \mid i \in \{1,2\}, n+2 \le j \le 2n+1\} \\
\cup\;& \{(t_{(1,2n+2)} \to \lambda, X \to Y), (t_{(2,2n+2)} \to \lambda, X \to Y)\} \\
\cup\;& \{(Y \to t), (Y \to v_0)\} \\
\cup\;& \{(v_i \to v_{i+1}, a_{i+1} \to \#) \mid 0 \le i \le m-2\} \cup \{(v_{m-1} \to \lambda, a_m \to \#)\};
\end{aligned}$$

Results II

P systems with chained rules when all the vectors contain either non-cooperative rules or catalytic rules (with only one catalyst) generate the class of all recursively enumerable sets of natural numbers.

$$NOPCRP_2(cat_1) = NRE.$$

- proved by a simulation of an arbitrary register machine.

Results II

• for each instruction $l_1 : (add(r), l_2, l_3) \in \mathcal{P}$ we add to R_2 the rules:
$(l_1 \to a_r l_2)$
$(l_1 \to a_r l_3)$

• for each instruction $l_1 : (sub(r), l_2, l_3) \in \mathcal{P}$ we add to R_2 the rules:
$(l_1 \to \overline{l_1} R_{l_1}, ca_r \to c, \overline{l_1} \to \overline{l_2}, \overline{l_2} \to l_2)$,
$(R_{l_1} \to \overline{R_{l_1}}, \overline{R_{l_1}} \to \lambda, \overline{l_1} \to l_3)$.

• for each instruction $l_h : (halt) \in \mathcal{P}$ we add to R_2 the rules:
$(l_h \to \lambda, a_1 \to a_{1out})$

Promoters vs Chained Rules

According with their definition promoters act globally and even one promoter might trigger several rules. If one considers the multiset $a^m b^n p$ with $m, n \geq 1$ and the rules

$$r_1 : a \to \alpha|_p \qquad r_2 : b \to \beta|_p$$

then both rules will be applied.

It might seem that in a vector $(p \to p, a \to \alpha)$ the execution of the rule $p \to p$ might "activate" in the next computation step the rule $a \to \alpha$ (hence it seems that the promoted rule $r_1 : a \to \alpha|_p$ can be easily simulated). What happens when one has to simulate the execution of a couple of rules (as r_1 and r_2), because simply considering in addition the vector $(p \to p, b \to \beta)$ will not solve the problem (recall that there is only one object p, hence either the rule $a \to \alpha$ or $b \to \beta$ will be executed in the second step).

Open Problems and New Ideas

$$\bullet NOPCRP_1(ncoo) = NET0L?$$

Let R be the set of rules from one given region of the P system and let $\mathbf{R}_{R\cup\{d\}}$ the set of all regular expression over the set of labels of $R \cup \{d\}$ (d comes from delay and means that in this step no rule is applied). Then, instead of sets of vectors of rules one can use sets of regular expressions to characterize the computation.

Thank You!