Spiking Neural P Systems with Inhibitory Axons

Rudolf FREUND Marion OSWALD
Faculty of Informatics
Vienna University of Technology
Vienna, Austria
rudi@emcc.at marion@emcc.at

Abstract

We extend the original model of spiking neural P systems by adding inhibitory axons. We show how computational completeness can already be obtained with extended spiking neural P system with inhibitory axons, and we also exhibit that finite extended spiking neural P systems with inhibitory axons characterize the regular sets. As a specific application example, we show how logical gates can be modelled with a static simple variant of (extended) spiking neural P systems with inhibitory axons.

1 Introduction

Based on the biological background of neurons sending electrical impulses (also called *spikes*) along axons to other neurons, in the area of neural computation several new models have been introduced, e.g., see [4], [6], [7]. P systems (membrane systems) have been introduced as a formal model implementing the hierarchical structure of membranes in living organisms and the biological processes in and between cells (an introduction to this field can be found in [9], for the actual state of the art in this area see [12]). Just recently, combining the ideas of P systems and spiking neurons, this led to a new variant of tissue P systems (see [3]) called spiking neural P systems, e.g., see [5], [10]. An extended version of spiking neural P systems allowing to send different informations along the axons between two neurons was investigated in [1].

In spiking neural P systems (see [5]), the contents of a neuron consists of a number of so-called spikes. The rules assigned to a cell allow us to send information to other neurons in the form of electrical impulses – spikes – which are summed up at the target cell; the application of the rules depends on the contents of the neuron. As inspired from biology, the cell sending out spikes may be "closed" for a specific time period corresponding to the refraction period of a neuron; during this refraction period, the neuron is closed for new input and cannot get excited ("fire") for spiking again. In [1], an extended version of this original model of spiking neural P systems was introduced based on some other observations from biology: for example, the length of the axon may cause a time delay before a spike arrives at the target; moreover, the spikes coming along different axons may cause effects of different magnitude.

Another quite natural feature found in biology and also used in the area of neural computation is that of inhibitory neurons or connections between neurons. Hence, in this paper we consider spiking neural P systems with inhibitory axons, thus extending again the model of extended spiking neural P systems by considering inhibitory axons that allow for "closing" a neuron for one step by sending a spike along such an inhibitory axon to this neuron from another one.

The rest of the paper is organized as follows: In the next section, after giving some preliminary definitions, we introduce the model of extended spiking neural P system with inhibitory axons. In Section 3, we exhibit some theoretical results for this new model of extended spiking neural P system with inhibitory axons: finite spiking neural P systems with inhibitory axons (where the number of spikes that may be stored in each neuron can be bounded) characterize the regular sets in the same way as the well-known McCulloch-Pitt networks (e.g., see [8]); without any bounds on the number of spikes in the neurons we obtain computational completeness as already shown for the other models of spiking neural P systems. As a specific application, in Section 4 we show how spiking neural P systems with inhibitory axons can be used to specify/simulate logical gates. A short summary and an outlook to future research conclude the paper.

2 Definitions

For the basic elements of formal language theory needed in the following, we refer to any monograph in this area, e.g., to [2] and [11]. We just list a few notions and notations: V^* is the free monoid generated by the alphabet V under the operation of concatenation and the empty string, denoted by λ, as unit element; for any $w \in V^*$, $|w|$ denotes the number of symbols in w (the *length* of w). $\mathbb{N}_+$ denotes the set of positive integers (natural numbers), $\mathbb{N}$ is the set of non-negative integers, i.e., $\mathbb{N} = \mathbb{N}_+ \cup \{0\}$. The interval of non-negative integers between k and m is denoted by $[k..m]$. For any $k \in \mathbb{N}$, $RE\left(\mathbb{N}^k\right)$ and $REG\left(\mathbb{N}^k\right)$ denote the sets of recursively enumerable and regular subsets of $\mathbb{N}^k$, respectively. $REG(\{a\})$ denotes the set of regular languages over the alphabet $\{a\}$ (observe that there is a one-to-one correspondence between the sets in $REG(\{a\})$ and the sets in $REG(\mathbb{N})$).

Extended spiking neural P systems with inhibitory axons

For the motivation and the biological background of spiking neural P systems we refer the reader to [5].

An *extended spiking neural P system with inhibitory axons* is a construct

$$\Pi = (m, S, R, F)$$

where

- m is the number of *cells* (or *neurons*); the neurons are uniquely identified by a number between 1 and m (obviously, we could instead use an alphabet with m symbols to identify the neurons);
- S describes the *initial configuration* by assigning an initial value (of spikes) to each neuron;
- R is a finite set of *rules* of the form $\left(i, E/a^k \to P, d\right)$ such that $i \in [1..m]$ (specifying that this rule is assigned to cell i), $E \subseteq REG(\{a\})$ is the *checking set* (the current number of spikes in the neuron has to be from E if this rule shall be executed), $k \in \mathbb{N}$ is the "number of spikes" (the energy) consumed by this rule, P is a (possibly empty) set of *productions* of the form (l, w) where $l \in [1..m]$ (thus specifying the target cell), and $w = \overline{a}$ (we also call $\overline{a}$ *inhibitor*) or $w \in \{a\}^*$ is the *weight* of the energy sent along the axon from neuron i to neuron l, and d is the delay.
- $F \subseteq [1..m]$ specifies the set of neurons which store the output.

Starting from the initial configuration of the system that is given by S, a *transition* from one configuration to another one now works as follows: for each neuron i, we first check whether we find an applicable rule $\left(i, E/a^k \to P, d\right)$ (i.e., the number of spikes in neuron i coincides with the regular checking set E). If this is the case and the neuron is not blocked due to the delay of a previously applied rule, then the neuron "fires", i.e., for every production (l, w) occurring in the set P the corresponding package (l, w) is sent from i to neuron l; if $d > 0$, then the neuron is blocked for the next d steps, i.e., it cannot apply another rule and, moreover, all inputs arriving during the next $d - 1$ steps are ignored. Now for every neuron we have to consider the following two cases:

- If in any of the packages just having arrived in a neuron we find an inhibitor $\overline{a}$, then neuron l is blocked for one step, i.e., no rule can be applied in neuron l and no input from other cells is taken in this step.
- On the other hand, if in the packages just having arrived in a neuron we find no inhibitor, for the other packages (l, w) with $w \in \{a\}^*$, the weight w in such package is added to the corresponding number of spikes in neuron l (provided the neuron is not closed for input).

A *computation* is a sequence of configurations starting with the initial configuration given by S. The result of a halting computation (where no neuron is blocked, no rule can be applied anymore) then can be found in the output neurons specified by F.

Note that in the system defined above, we did not introduce any delay for the packages along the axons, as e.g. done in [1]. The original version of spiking neural P systems as defined in [5] corresponds with a very restricted variant of extended spiking neural P system with inhibitory axons where we do not use inhibitors and, moreover, the number of spikes sent from a neuron i to others is always fixed and (i) either no spikes are emitted, which corresponds to the case $P = \{\ \}$ in the rule $\left(i, E/a^k \to P\right)$ – such a rule is called a *forgetting rule* –, or (ii) the rule is of the form $\left(i, E/a^k \to \{(l, a) \mid l \in M_i\}\right)$ for some $M_i \subseteq [1..m]$ only depending on i – such a rule is called a *spiking rule*. On the other hand, this variant can easily be extended to *spiking neural P systems with inhibitory axons* by also allowing rules of the form

$$\left(i, E/a^k \to \{(l, a) \mid l \in M_i\} \cup \{(l, \overline{a}) \mid l \in N_i\}\right)$$

for some $N_i \subseteq [1..m]$ only depending on i, with $M_i \cap N_i = \{\ \}$.

If we only allow the rules to be of the form $\left(i, E/a^k \to P\right)$ with $E = \left\{a^l\right\}$ for some $l \geq 1$ and with all productions (l, w) being of the form $w = a^j$ for some $j \geq 0$ or $w = \bar{a}$, then such an extended spiking neural P system with inhibitory axons is called *finite.*

3 Theoretical Results

Extended spiking neural P systems without inhibitory axons were shown to be computationally complete [1] even if not using any delay in the neurons or axons, which also proves the computational power of the systems introduced in this paper. Hence, we obtain the following result:

Theorem 1. For every set L in $RE\left(\mathbb{N}^k\right)$, we can construct an extended spiking neural P systems with inhibitory axons Π generating L.

It remains a challenging research topic for future theoretical investigations whether inhibitory axons could be used to obtain computational completeness with various restrictions thereby trading inhibitory axons for some other features like forgetting rules.

Finite extended spiking neural P systems with inhibitory axons (where we only allow the rules to be of the form $\left(i, E/a^k \to P\right)$ with $E = \left\{a^l\right\}$ for some $l \geq 1$ and with all productions (l, w) being of the form $w = a^j$ for some $j \geq 0$ or $w = \bar{a}$) can only generate regular sets, because the number of spikes to be stored in each neuron can be bounded (following the construction given in [1] for extended spiking neural P systems), i.e., we obtain the following result:

Theorem 2. Finite extended spiking neural P systems (with inhibitory axons) characterize $REG\left(\mathbb{N}^k\right)$.

4 Simulating Logical Gates

In this section we restrict ourselves to *spiking neural P systems with inhibitory axons,* i.e., all the rules are of one of the following forms:

- $\left(i, E/a^k \to \{\ \}\right)$; such a rule is called a *forgetting rule*;
- $\left(i, E/a^k \to \{(l, a) \mid l \in M_i\} \cup \{(l, \overline{a}) \mid l \in N_i\}\right)$ for some $M_i, N_i \subseteq [1..m]$, $M_i \cap N_i = \{\ \}$, only depending on i; such a rule is called a *spiking rule.*

The restriction to these kinds of rules allows us to represent such a *spiking neural P system with inhibitory axons* by a directed graph as follows:

- the neurons are represented by the nodes of the graph;
- the spiking rules and the forgetting rules are specified in the nodes;
- for each $j \in M_i$ we draw a directed edge from i to j and mark it as *excitatory* edge;
- for each $j \in N_i$ we draw a directed edge from i to j and mark it as *inhibitory* edge.

Using spiking neural P systems with inhibitory axons, we now can easily represent (simulate) logical gates. For example, the simulation of a NAND-gate by a spiking neural P systems with inhibitory axons can be seen in Figure 1; it corresponds to a system

$$(3, \{(1, a), (2, \lambda), (3, \lambda), R, \{3\}\})$$

with R containing the rules
$(1, \{a^2\}/a \to \{(3, \overline{a})\})$, $(1, \{a\}/a \to \{\ \})$,
$(2, \{a\}/a \to \{(2, a), (3, a)\})$, and
$(3, \{a\}/a \to \{(out, a)\})$ – *out* specifies that the signal coming from neuron 3 is the output of the system, but it can be taken as input signal for another NAND-system.

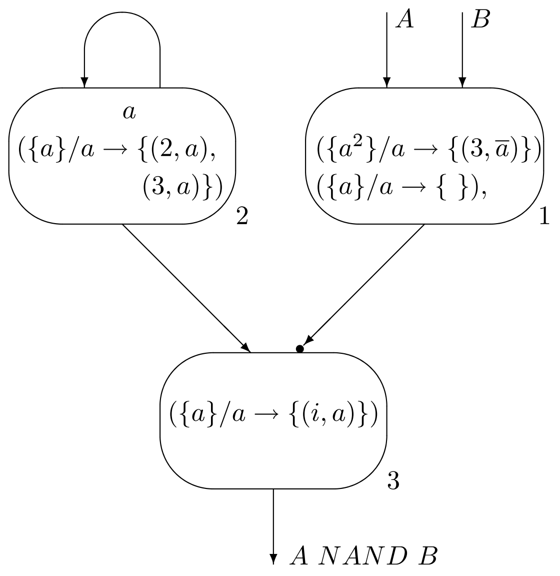
A
B
a
$(\{a\}/a \to \{(2, a), (3, a)\})$
2
$(\{a^2\}/a \to \{(3, \overline{a})\})$
$(\{a\}/a \to \{\ \})$,
1
$(\{a\}/a \to \{(i, a)\})$
3
A NAND B

Figure 1: NAND-gate.

In each step, neuron 2 sends one spike to neuron 3. Neuron 1 has two input axons (marked by A and B).

If a spike arrives from only one of these axons, then the rule $(\{a\}/a \to \{\})$ is executed, and the output neuron 3 spikes. Only if neuron 1 gets a spike from both its input axons, then the rule $(\{a^2\}/a \to \{(3,\overline{a})\})$ will be applied, which now inhibits neuron 3 from firing because the *inhibitory axon* (indicated by a dot at the end of the directed edge) from neuron 1 to neuron 3 is activated.

As a specific feature of the spiking neural P system with inhibitory axons described above, we can see that for every neuron j in this system, either M_j or N_j is empty.

As is well known, every boolean function can be represented by just using NAND-gates. Hence, combining such systems as described above, we can easily represent every boolean function by a corresponding spiking neural P system with inhibitory axons. In this case, the answer to an input arrives in $2k$ steps where k is the depth of the logical network of NAND-gates representing the given function. Moreover, in this case we do not consider halting computations, yet instead observe the spike train (the sequence of zeroes and ones in the output neuron) taking into account the delay of $2k$.

5 Conclusion

We have introduced the model of extended spiking neural P systems with inhibitory axons. Based on the theoretical results already proved in [1] and [5], we have exhibited that finite extended spiking neural P systems with inhibitory axons characterize the regular sets; on the other hand, already very restricted variants of extended spiking neural P systems with inhibitory axons allow us to obtain computational completeness, i.e., to characterize the recursively enumerable sets.

Spiking neural P systems with inhibitory axons can be used to specify logical gates, but they also promise to be interesting for specifying other models of computation; for example, in the future we shall also investigate the relation between extended spiking neural P systems with inhibitory axons and Petri nets.

Acknowledgements

The work of Marion Oswald is supported by FWF-project T225-N04.

References

[1] A. Alhazov, R. Freund, M. Oswald, M. Slavkovik, Extended Spiking Neural P Systems Generating Strings and Vectors of Non-Negative Integers, in H. J. Hoogeboom, Gh. Paun, G. Rozenberg (Eds.), Pre-proceedings of the 7th Workshop on Membrane Computing WMC7, 88-101 (2006)

[2] Dassow J, Păun Gh (1989) Regulated Rewriting in Formal Language Theory. Springer, Berlin

[3] Martín-Vide C, Pazos J, Păun Gh, Rodríguez-Patón A (2002) A new class of symbolic abstract neural nets: Tissue P systems. In: Proceedings of COCOON 2002, Singapore, Lecture Notes in Computer Science 2387, Springer-Verlag, Berlin, 290–299

[4] Gerstner W, Kistler W (2002) Spiking Neuron Models. Single Neurons, Populations, Plasticity. Cambridge Univ. Press

[5] Ionescu M, Păun Gh, Yokomori T (2006) Spiking neural P systems. Fundamenta Informaticae 71, 2–3:279–308

[6] Maass W (2002) Computing with spikes. Special Issue on Foundations of Information Processing of TELEMATIK 8, 1:32–36

[7] Maass W, Bishop C (eds) (1999) Pulsed Neural Networks. MIT Press, Cambridge

[8] Minsky M L (1967) Computation: Finite and Infinite Machines. Prentice Hall, Englewood Cliffs, New Jersey

[9] Păun Gh (2002) Computing with Membranes: An Introduction. Springer, Berlin

[10] Păun Gh, Pérez-Jiménez MJ, Rozenberg G (2006) Spike trains in spiking neural P systems, Intern J Found Computer Sci, to appear (also available at [12])

[11] Rozenberg G, Salomaa A (eds) (1997) Handbook of Formal Languages (3 volumes). Springer, Berlin

[12] The P Systems Web Page, `http://psystems.disco.unimib.it`