Neurocomputing 151 (2015) 1439–1445

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Asynchronous spiking neural P systems with rules on synapses

Tao Song [a], Quan Zou [b], Xiangrong Liu [b], Xiangxiang Zeng [b],*

[a] Key Laboratory of Image Processing and Intelligent Control, School of Automation, Huazhong University of Science and Technology, Wuhan 430074, Hubei, China
[b] Department of Computer Science, Xiamen University, Xiamen 361005, Fujian, China

A R T I C L E I N F O

Article history:
Received 16 May 2014
Received in revised form
12 October 2014
Accepted 13 October 2014
Communicated by Long Cheng
Available online 6 November 2014

Keywords:
Membrane computing
Neural computing
P system
Spiking neural P system
Universality

A B S T R A C T

Spiking neural P systems (shortly called SN P systems) are a group of neural-like computing models inspired by the functioning of spiking neurons. Recently, a new variant of SN P systems, called SN P systems with rules on synapses (RSSN P systems for short) were proposed. In such systems, at any moment the synapses with enabled rules should use one of the enabled rules, and all the synapses work in the synchronous manner. In this work, we consider RSSN P systems with a non-synchronized (i.e., asynchronous) use of rules: in any step, the synapses with enabled rules are not obligated to use the enabled rules. It is proved that the non-synchronization does not decrease the computing power in the case of using extended rules (several spikes can be produced for once spiking). We obtain again the equivalence with Turing machines (interpreted as generators of sets of natural numbers). Moreover, we construct a universal asynchronous RSSN P system with 94 neurons, which can compute any Turing computable function.

1. Introduction

Natural computing is the field of research that investigates human-designed computing inspired by nature. It investigates models and computational techniques inspired by nature, as well as the information processing, phenomena taking place in nature [1]. Membrane computing, initiated by Păun [2], is a new branch of natural computing, whose aim is to abstract computing ideas from the structure and the functioning of a single cell and from complexes of cells, such as tissues, organ and human brain, in order to construct computing models. The models obtained are distributed and parallel computing devices, usually called P systems. In the present work, we deal with a class of neural-like membrane computing models, named spiking neural P systems (shortly called SN P systems) [3].

SN P systems are a class of distributed and parallel computing models inspired by spiking neurons, which are currently much investigated in neural computing, see, e.g., [4,5]. SN P systems can be defined in a symbolic way, but in most previous works, they were represented in a graphic way. Generally, an SN P system can be represented by a directed graph, where neurons are the nodes and the synapses are directed edges. We use a rounded rectangle to represent a neuron, which may contain a number of spikes, spiking (or firing) rules and forgetting rules. By using the firing rule, a neuron can send information (in form of spikes) to its neighboring neurons. The spikes from different neurons can be accumulated at the target neurons. By using the forgetting rule, a predefined number of spikes will be removed from the neuron, thus are removed out of the system. One neuron in the system is specified as the output neuron, which can emit spikes to the environment. The computing result of a computation is encoded by the time span elapsed between the first two consecutive spikes sent into the environment by the (output neuron of the) system.

Until now, many variants of SN P systems, including especially their computational properties have been investigated. Some variants of SN P systems have been proposed and proved to be Turing universal as number accepting or generating devices, such as sequential SN P systems [6], asynchronous SN P systems [7], asynchronous SN P systems with local-synchronization [8], SN P systems with astrocyte-like control [9,10], SN P systems with anti-spikes [11], time-free SN P systems [12] and homogenous SN P systems [13]; as language generators, that is, characterizing recursively enumerable languages, see e.g. [14–16]; and computing any Turing computable functions [17–20]. SN P systems with cell division or neuron budding can generate (even exponential) working space during the computation (in linear time), thus provide a way to (theoretically) solve computational hard problems, such as SAT, in feasible (polynomial or even linear) time [21–23].

* Corresponding author.
E-mail addresses: taosong@mail.hust.edu.cn (T. Song), zouquan@xmu.edu.cn (Q. Zou), xrliu@xmu.edu.cn (X. Liu), xzeng@xmu.edu.cn (X. Zeng).

http://dx.doi.org/10.1016/j.neucom.2014.10.044
0925-2312/© 2014 Elsevier B.V. All rights reserved.

Recently, a new variant of SN P systems, called SN P systems with rules on synapses (RSSN P systems for short), were proposed in [24]. In these systems, the neurons contain only spikes, and the spiking and forgetting rules are moved onto the synapses. At any step, if the number of spikes in a given neuron is "recognized" by a rule on a synapse leaving from that neuron, then the rule becomes enabled. In this case, a number of spikes are consumed in the neuron and a number of spikes are sent to the neuron at the end of the synapse. At any moment, if several synapses starting in the same neuron have rules that can be applied, then all enabled rules will consume the same number of spikes from the given neuron. By using the rules, the neuron consumes the exact number of spikes requested by one of the rules. All the synapses use their rules nondeterministically in a synchronized manner, that is, at any moment all the synapses with enabled rules should use one of its enabled rules. For each synapse, the system works in a sequential way, that is, in each step at most one rule can be applied at any synapse. RSSN P systems have been proved to be universal with simple topological structures [24].

In the proof of the universal result, the synchronization plays a crucial role, but both from a mathematical point of view and from a neuro-biological point of view it is rather natural to consider non-synchronized systems. In this work, we investigate asynchronous RSSN P systems, where the use of rules on any synapse is not obligatory. Specifically, in a given time unit, even if a synapse has a rule enabled, this rule is not obligatorily applied. The neuron that the synapse starts from may receive spikes from the neighboring neurons. If the unused rule on the synapse can be applied later, it is applied later, without any restriction on the interval during which it has remained unused. If the new spikes make the rule non-applicable, then the computation will continue in the new circumstances (maybe other rules are enabled now).

The synapse starting from the output neuron also uses its rules in an asynchronous manner, so the distance in time between the spikes sent out by the system is no longer relevant. Hence, for asynchronous RSSN P systems, the result of the computation is defined as the total number of spikes sent out to the environment. This makes it necessary to consider only halting computations (The computations which do not halt are ignored and provide no output.)

By simulating the register machine working in generating modes, it is obtained that asynchronous RSSN P systems are universal as number generators in case of using extended rules. In addition, an asynchronous RSSN P system with 94 neurons is obtained which can compute any Turing computable function. The result has a nice interpretation that loss in power entailed by removing the synchronization from ASN P systems can be compensated by using more neurons.

The rest of this paper is organized as follows. In the next section, we introduce the definition of asynchronous RSSN P systems. Section 3 gives an example to show how asynchronous RSSN P systems work. In Section 4, we prove the universality of asynchronous RSSN P systems. Conclusions and remarks are drawn in Section 5.

2. Asynchronous RSSN P systems

In this section, asynchronous RSSN P system are introduced. We specify the sets of numbers generated by such systems, as well as the way to compute functions using the system.

It is useful for readers to have some familiarity with basic elements of formal language theory, e.g., from [25], as well as basic concepts and notions in SN P systems [3]. We introduce the some basic prerequisites.

For an alphabet Σ, Σ^* denotes the set of all finite strings of symbols from Σ, the empty string is denoted by λ, and the set of all nonempty strings over Σ is denoted by Σ^+. When $\Sigma = \{a\}$ is a singleton, then we write simply a^* and a^+ instead of $\{a\}^*, \{a\}^+$.

An asynchronous RSSN P system of degree $m \geq 1$ is a construct of the form

$\Pi = (O, \sigma_1, \sigma_2, \ldots, \sigma_m, syn, i_{out})$, where

- $O = \{a\}$ is the singleton alphabet (a is called a *spike*);
- $\sigma_1, \sigma_2, \ldots, \sigma_m$ are *neurons* of the form $\sigma_i = (n_i)$ with $1 \leq i \leq m$, where n_i is initial number of spikes in neuron σ_i;
- *syn* is the set of synapses, and each element in *syn* is a pair of the form $((i,j), R_{(i,j)})$, where (i,j) indicates there is a synapse connecting neurons σ_i and σ_j with $i,j \in \{1,2,\ldots,m\}, i \neq j$, and $R_{(i,j)}$ is a set of finite rules on synapse (i,j) of the following two forms:
  (1) $E/a^c \to a^p$, where E is a regular expression over O and $c \geq p \geq 1$;
  (2) $a^s \to \lambda$, for some $s \geq 1$, with the restriction that $a^s \notin L(E)$ for any rule $E/a^c \to a^p; d$ from any rule $R_{(i,j)}$;
- i_{out} indicates the *output* neuron (sending spikes to the environment).

A rule $E/a^c \to a^p$ with $p \geq 1$ is called *extended spiking rule* or *extended rule*; a rule $E/a^c \to a^p$ with $p=0$ is written in the form $E/a^c \to \lambda$ and is called a *forgetting rule*. If $L(E) = \{a^c\}$, then the rules are written in the simplified form $a^c \to a^p; d$ and $a^c \to \lambda$. A rule of the type $E/a^c \to a$ and $a^c \to \lambda$ is said to be *standard*.

The spiking rules are applied as follows. If $E/a^c \to a^p; d \in R_{(i,j)}$ and neuron σ_i contains k spikes such that $a^k \in L(E)$, $k \geq c$, then the rule is enabled. For synapse (i,j), the enabled rule is not obligatorily applied, that is the synapse may choose to remain unfired without using any enabled rule or fire by using the enabled rule.

If the synapse chooses to fire by using the rule, c spikes are consumed from neuron σ_i and p spikes are sent to neuron σ_j. (We have the restriction that at any moment, if several synapses starting in the same neuron have rules that can be applied, then all enabled rules consume the same number of spikes from the given neuron. Specifically, at any moment the applied rules on the synapses leaving from neuron σ_i should have the form $E_u/a^{c_u} \to a^p$; and then by using the rules c spikes are consumed from neuron σ_i.) (In [24], there is assumption that at any moment, all enabled rules on the synapses starting from the same neuron should consume the same number of spikes. This assumption was proposed by considering the fact that at any moment, any neuron with enabled rules on synapses starting from it will fire by consuming a fixed number of spike, and to ensure all the enabled rules can be applied, it was forced that all the enabled rules should have a uniform regular expression (denoted by E_u) and consume the same number of spikes (denoted by c_u).

If the synapse chooses to remain unfired, no enabled rule on the synapse can be applied. The neuron that the synapse starts from may consume a number of spikes by using enabled rule(s) on other synapse(s), or consume no spike in case that all the synapses starting from it keeps unfired. No matter whether there exists synapses firing or not, the neuron can receive spikes from the neighboring neurons. If the unused rules on the synapses may be applied later, they will be applied later, without any restriction on the interval when it has remained unused. If the new coming spikes made the unused rule non-applicable, then the computation continues in the new circumstances (maybe other rules are enabled now).

As usual in SN P systems, a global clock is assumed, marking the time for all neurons and synapses. It is possible that more than one rule can be applied on a synapse in any step, since two spiking