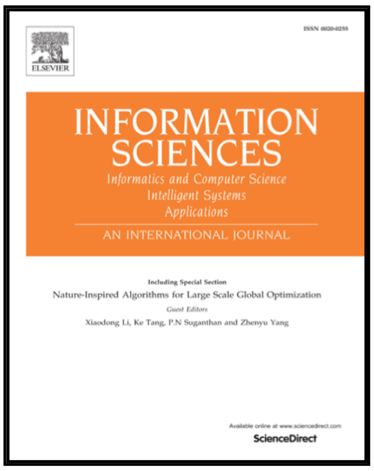

Design of Logic Gates Using Spiking Neural P Systems with Homogeneous Neurons and Astrocytes-like Control

Tao Song, Pan Zheng, M.L. Dennis Wong, Xun Wang

PII: S0020-0255(16)30624-7
DOI: 10.1016/j.ins.2016.08.055
Reference: INS 12458

To appear in: *Information Sciences*

Received date: 4 September 2015
Revised date: 8 August 2016
Accepted date: 15 August 2016

Design of Logic Gates Using Spiking Neural P Systems with Homogeneous Neurons and Astrocytes-like Control

Tao Song[a,b], Pan Zheng[b], M. L. Dennis Wong[b], Xun Wang[*,a]

[a]*College of Computer and Communication Engineering, China University of Petroleum, Qingdao 266580, Shandong, China*
[b]*Faculty of Engineering, Computing and Science, Swinburne University of Technology Sarawak Campus, Kuching 93350, Malaysia*

Abstract

In biological nervous systems, the operation of interacting neurons depends largely on the regulation from astrocytes. Inspired by this biological phenomenon, spiking neural P systems, i.e. SN P systems, with astrocyte-like control were proposed and were proven to have "Turing completeness" as computing models. In this work, the application of such systems for creating logical operators is investigated. Specifically, it is obtained in a constructive way that SN P systems with astrocyte-like control can synthesize the operations of Boolean logic gates, i.e. AND, OR, NOT, NOR, XOR and NAND gates. The resulting systems are simple and homogeneous, which means only one type of neuron with a unique spiking rule is used. With these neural-like logic gates, more complex Boolean circuits with cascade connections can be constructed. As such, they can be used to implement finite computing devices, such as the finite transducers. These results demonstrate a novel method of constructing logic circuits that work in a neural-like manner, as well as shed some lights on potential directions of designing neural circuits theoretically.

Key words: Bio-inspired computing, Spiking Neural P Systems, Astrocytes-like control, Logic gates, Boolean circuits

*Corresponding author. Email: wangxun0830@hotmail.com

1. Introduction

In biological nervous systems, the processing and transmission of information among the neurons had inspired and influenced computer scientists in designing powerful computational devices and intelligent machines for the past decades. In particular, the study of neural computing models based on spiking function is an emerging branch of natural computing which has been explored in great depth [19, 23, 24]. In 2006, a new class of neural computing models, namely spiking neural P systems (shortly named SN P systems) were proposed [16] through modeling the way neurons communicate via electrical impulses (spikes). Such systems are also known as a specific group of ingredients in membrane computing [29], and corresponding to a paradigm shift from cell-like to neural-like architectures.

An SN P system is essentially a networked model constructed with a number of neurons placed in the nodes of a directed graph. The neurons are able to transmit signals, i.e. spikes, to one other along the synapses, i.e. the arcs of the directed graph. The neurons then process the information received in the form of spikes by spiking (firing) and forgetting rules. The execution of a spiking rule is triggered by matching the number of spikes in a neuron with the regular expression associated with the rule. Using a spiking rule, a neuron can send a number of spikes to its neighbouring neurons, and the spikes are accumulated at the target neuron. With the execution of a forgetting rule, a predefined number of spikes will be removed from a neuron. In this system, the input neurons, act like the biological sensory neurons, receive spikes (information) in the nervous system, and the output neuron sends spikes to the external environment.

Taking ideas from various neurobiological phenomena, there have been several variants of SN P proposed, such as asynchronous SN P systems [5], asynchronous SN P systems with local synchronization [42], SN P systems with anti-spikes [20, 27], sequential SN P systems [15, 37, 53], SN P systems with rules on synapses in equal consumption mode [38], maximum spiking mode [39] and maximum spike consumption mode [40], SN P systems with plastic structure and self-organization [3, 49], SN P systems with request rules [41], cell-like SN P systems [50]. Most of these variants are Turing universal as number generating/accepting devices, language generating devices or function computing devices. There are some significant applications of SN P systems in solving issues in practical engineering and science fields, such as fuzzy reasoning SN P systems for fault diagnosis [33, 47, 48], fuzzy SN P systems for knowledge representation [45, 46], SN P systems for approximately solving combinatorial optimization problems [52], SN P systems based nuclear signal export identifying method [7] and a theoretical

CPU model made of SN P systems [12].

In biological nervous systems, apart from the neurons, astrocytes are described as star-shaped glial cells spanning around neurons. One can refer to [1, 34] for the biological details of the interaction between neurons and astrocytes.

Through studying the "excitatory and inhibitory" behaviours of the astrocytes during the interaction among neurons, SN P systems with astrocyte-like control were proposed in [32]. In the systems, each astrocyte is associated with a pre-defined threshold, and is able to sense the number of spikes passing through its neighbouring synapses. At any moment, if the number of spikes passing through the neighbouring synapses of an astrocyte is greater than or equal to its threshold, the astrocyte will then have an inhibitory influence to remove these spikes passing along the synapses. On the other hand, if the number of spikes passing through the neighbouring synapses of an astrocyte is strictly less than its threshold, the astrocyte will then have an excitatory influence on the neighbouring synapses, i.e. the spikes can "safely" pass to the destination neurons. SN P systems with astrocyte-like control have been proven to be Turing universal as number computing devices [25]. It was suggested in [32] that finding some applications of SN P systems with astrocytes-like control is of great interest to the research community.

Bio-inspired computing models based Boolean gates and Boolean circuits have been heavily investigated, see e.g. membrane computing systems based logic gates in [6, 11, 17] and DNA computing models based logic gates in [21, 43, 44]. With DNA strand displacement strategy, the so-called "DNA neuron" was developed to perform logic computation in nano-scale, but these "DNA neurons" sometimes cannot achieve the correct logic computation result since there are several spiking rules in a neuron, owing to the additional challenge that the proposed system used only the units of volume of DNA molecules to represent the stack of spikes [43]. In order to construct neural-like logic gates and circuits, SN P systems are used to design logic gates in [18], where it needs several types of neurons to simulate a logic gate. In [26], anti-spikes were introduced into SN P system to design NAND gate where by three types of neurons were needed to simulate a NAND gate, and the neuron had at least two spiking rules. In 2010, Metta et. al. [26] suggested a possible research direction to design logic gates and circuits using some other variants of SN P systems.

Pioneering works of designing logic gates by simple neurons was proposed in [2, 54], which constructed NAND gates by SN P systems with simple neurons and excitatory/inhibitory astrocytes. In this work, this result is further improved by creating Boolean logic gates with only inhibitory astrocytes-like control, simple and homogenous neurons. In particular, we look into the construction logic AND, OR, NOT, NOR, XOR and NAND gates by SN P systems with a unique type of neuron. The resulting systems are simple and homogeneous, which means each neuron in the systems has the same and unique spiking rule that makes each neuron to function like a "transmitter of information", yet no forgetting rule is formulated. This arrangement leads to a new class of simple and homogeneous SN P systems. After that, an example of Boolean circuit is achieved by the logic gates of SN P systems and a synchronization module for making n steps delay. As such, we demonstrate that SN P systems with astrocyte-like control are capable of emulating finite computing devices (based on logic gates), such as the finite transducers. These results can be extended to a novel way for constructing logic circuits working in a neural-like manner, as well as offer some potential directions to design neural circuits theoretically.

The merits of our results are: 1) the individual neuron is simple in structure but universal, i.e. only one type of neuron is used for constructing any layout of Boolean circuits; and 2) with the modulation of astrocytes and cooperating with each other, a network of neurons and astrocytes can perform complex functions.

2. Spiking Neural P Systems with Astrocyte-like Control

Some necessary prerequisites of basic concepts and notions of formal language theory from [35] are firstly recalled. For an alphabet set V, V^* denotes the set of all finite strings of symbols from V, the empty string is denoted by λ, and the set of all nonempty strings over V is denoted by V^+. If $V = \{a\}$ is a singleton, we write simply a^* and a^+ instead of $\{a\}^*$ and $\{a\}^+$.

A regular expression over an alphabet V is defined as follows: (i) λ and each $a \in V$ are regular expressions, (ii) if E_1, E_2 are regular expressions over V, it holds that $(E_1)(E_2)$, $(E_1) \cup (E_2)$, and $(E_1)^+$ are regular expressions over V, and (iii) nothing else is a regular expression over V. With each regular expression E, it is associated a language $L(E)$, defined in the following way: (i) $L(\lambda) = \{\lambda\}$ and $L(a) = \{a\}$, for all $a \in V$, (ii) $L((E_1) \cup (E_2)) = L(E_1) \cup L(E_2)$, $L((E_1)(E_2)) = L(E_1)L(E_2)$, and $L((E_1)^+) = (L(E_1))^+$, for all regular expressions E_1, E_2 over V. Unnecessary parentheses can be omitted when writing a regular expression, and $(E)^+ \cup \{\lambda\}$ can also be written as E^*.

The following definition of SN P systems with astrocyte-like control is self-contained, but familiarity with the basic elements of classic SN P systems (from [30, 31]) is helpful.

An SN P system with astrocyte-like control of degree $m \geq 1$ is a construct as follows:

$$\Pi = (O, \sigma_1, \sigma_2, \dots, \sigma_m, syn, In, Out), \text{ where}$$

- $O = \{a\}$ is a singleton alphabet, and a is called *spike*;
- $\sigma_1, \sigma_2, \dots, \sigma_m$ are *neurons* of the form $\sigma_i = (n_i, R_i)$ with $1 \leq i \leq m$, where
  1. $n_i \in \mathbb{N}$ is the *initial number of spikes* contained in neuron σ_i;
  2. R_i is a finite set of *rules* of the following two forms:
     (a) $E/a^c \rightarrow a; d$, where E is a regular expression over O, $a^c \in L(E)$, $c \geq 1$ and $d \geq 0$;
     (b) $a^s \rightarrow \lambda$, for some $s \geq 1$, with the restriction that $a^s \notin L(E)$ for any rule $E/a^c \rightarrow a; d$ from R_i;
- $syn \subset \{1, 2, \dots, m\} \times \{1, 2, \dots, m\}$ with $(i, i) \notin syn$, is the set of *synapses* between neurons;
- $ast_1, ast_2, \dots, ast_l$ are *astrocytes*, of the form $ast_i = (syn_{ast_i}, t_i)$, where $1 \leq i \leq l$, $syn_{ast_i} \subseteq syn$ is the set of synapses controlled by astrocyte ast_i, and $t_i \geq 0$ is the *threshold* of astrocyte ast_i;
- $In \subseteq \{\sigma_1, \sigma_2, \dots, \sigma_m\}$ is the set of *input* neurons (whose element can read spikes from the environment), and $Out \subseteq \{\sigma_1, \sigma_2, \dots, \sigma_m\}$ indicates the set of *output* neurons (each can emit spikes into the environment).

Rules of the form $E/a^c \rightarrow a; d$ are named *spiking or firing rules*. At any moment, if neuron σ_i contains k spikes and $a^k \in L(E)$, $k \geq c$, then the spiking rule $E/a^c \rightarrow a; d$ is used, by which c spikes are consumed and one spike is produced. If $d = 0$, then the spike is emitted to each neuron σ_j such that $(i, j) \in syn$ immediately; if $d = 1$, then the spike is emitted one step later, etc. A global clock is timed for the neurons in the system. In general, if the rule is used at step t and $d \geq 1$, in steps $t, t+1, t+2, \dots, t+d-1$ the neuron is in "closed" status, which corresponds to the refractory period from neurobiology, so that it cannot receive new spikes and use any rule. If a neuron sends a spike to a closed neuron at any moment, that particular spike will be lost, i.e., removed from the system. In the step $t + d$, neuron σ_i becomes open again (it can receive spikes at this step and may fire again). For any spiking rule $E/a^c \rightarrow a; d$, if $L(E) = a^c$, the rule can be simplified as $a^c \rightarrow a; d$; in addition, if $d = 0$, it is simply written as $a^c \rightarrow a$. Since two spiking rules, $E_1/a^{c_1} \rightarrow a; d_1$ and $E_2/a^{c_2} \rightarrow a; d_2$ can have

$L(E_1) \cap L(E_2) \neq \emptyset$, it is possible that two or more spiking rules can be applied in a neuron, and in that case, only one of them is non-deterministically chosen to use. In each time unit, if neuron σ_i can use one of its rules, then a rule from R_i must be used on the tick of the clock, for all neurons at the same time.

Forgetting rules are of the form $a^s \rightarrow \lambda$ with $s \geq 1$. The forgetting rule is applied if and only if the neuron contains exactly s spikes and $a^s \notin L(E)$, for any E included in R_i, $i = 1, 2, \ldots, m$. By using the forgetting rule, s spikes will be removed out of the neuron. Note that in any neuron if a spiking rule is applicable, then no forgetting rule is enabled, and vice versa.

The rules are used sequentially at each neuron, at most one in each step, but the neurons work in parallel with another. It is important to take note that the applicability of a rule is determined by checking the total number of spikes contained in the neuron against a regular set associated with the rule.

The functioning of the astrocyte is to sense and control the spike traffic along its neighboring synapses. For an astrocyte $ast_i = (syn_{ast_i}, t_i)$, suppose that there are p spikes passing along the synapses from syn_{ast_i} at a certain moment. If $p \geq t_i$, astrocyte ast_i has an inhibitory function, that is, the p spikes are suppressed by astrocyte ast_i (removed out of the system). If $p < t_i$, astrocyte ast_i has an excitatory influence on the spikes passing along its neighboring synapses (the k spikes pass along the synapses and arrive at the destination neurons). It is possible that two or more astrocytes control the same synapse(s). In this case, if all these astrocytes have excitatory influence on the synapse, the spike along this synapse can pass to the destination neurons; if one of these astrocytes has an inhibitory influence on the synapses, then the spike along this synapse will be suppressed. Actually, in the systems constructed in this work, it never happens that two astrocytes control the same synapse(s).

At any moment, the *configuration* of the system is defined by the number of spikes in every neuron and the open-close status represented by the time delays to become open. It is mathematically denoted by $\langle r_1/t_1, r_2/t_2, \ldots, r_m/t_m \rangle$, where neuron σ_i contains $r_i \geq 0$ spikes and it will be open after $t_i \geq 0$ steps, $i = 1, 2, \ldots, m$; with this notation, the initial configuration is $\langle n_1/0, n_2/0, \ldots, n_m/0 \rangle$. Using the rules and astrocytes described above, one can define *transitions* among configurations. Any series of transitions starting from the initial configuration is called a *computation*. The system halts if it reaches a configuration where no rule can be applied in any neuron and all the neurons are open. The *result of a computation* is the total number of spikes sent into the environment by the output neuron, when the system halts.

A neuron with only one spiking rule is said to be *simple*. An SN P system is said to be simple if all neurons are simple. An SN P system is said to be *homogeneous* if the system consists of only one type of the neuron. In simple and homogeneous SN P systems, each neuron has the same and unique spiking rule.

In the following sections, SN P systems with astrocyte-like control are depicted graphically, which is easier to comprehend than they are described in mathematical symbols. A rounded rectangle with the initial number of spikes inside is used to represent a neuron. The rule in the neuron will be omitted, since all the neurons have the same spiking rule. The directed edges represent the synapses between neurons. A rhombus associated with a number (threshold value) graphically denotes an astrocyte, which have connections to its controlling synapses jointing on black dots. Each input/output neuron has an incoming/outgoing arrow, respectively, suggesting their communications with the environment.

3. Simulating Logic Gates

In this section, SN P systems with astrocyte-like control are constructed to emulate operations of logic gates AND, OR, NOT, NOR, XOR and NAND, respectively. In the systems, only one type of neuron is used, which has a unique spiking rule $a^*/a \to a$, so we omit the spiking rule in the neuron from the graphical forms of the systems for a better and clear illustration.

In SN P systems with astrocyte-like control, multiple input neurons are used to receive spikes from the environment, but only one output neuron is used to emit computation result into the environment. The inputs and outputs of logic gates, digits 0 and 1, are encoded by the numbers of spikes in SN P systems. In our design, if the input digit is 0, then the input neuron will receive one spike from the environment; if the input digit is 1, then input neuron will receive two spikes. At the output neuron of the system, if the number of the output spikes is one then it indicates that the computation result of the logic gate is 0. If the number of output spikes is two, then the result of the computation is 1. In other words, the binary digit 0 is encoded in one spike and the binary digit 1 is encoded in two spikes.

Theorem 3.1. *The logic AND gate can be emulated by a simple and homogeneous SN P system with astrocyte-like control having six neurons and one astrocyte.*

Proof An SN P system with astrocyte-like control Π_{AND}, shown in Figure 1, is constructed, which can emulate operation of a logic AND gate. The system Π_{AND} has two input neurons σ_{in_1} and σ_{in_2} to receive spikes from the environment, and one output neuron σ_{out} to emit the computation result. In the following, the four cases of inputs to logic ADD gate, 00, 01, 10, 11, are considered.

– If the inputs are $x_1 = 0, x_2 = 0$, each of neurons σ_{in_1} and σ_{in_2} receives one spike from the environment. With one spike, the two neurons fire by using the spiking rule $a^*/a \to a$. Neuron σ_{in_1} sends one spike to neurons σ_1 and σ_2, while neuron σ_{in_2} sends one spike to neurons σ_1 and σ_3. In this way, neuron σ_1 accumulates two spikes inside, and neurons σ_2 and σ_3 contain one spike. In the next step, neurons σ_1, σ_2 and σ_3 fire by using rule $a^*/a \to a$. Neuron σ_1 ends with one spike, and neurons σ_2 and σ_3 have no spike inside. Neurons σ_1, σ_2 and σ_3 send one spike to neuron σ_{out} along synapses $(1, out)$, $(2, out)$ and $(3, out)$, respectively. Astrocyte ast_1 controls the three synapses with threshold value 2, so the three spikes will be suppressed. With one spike, neuron σ_1 fires for the second time by rule $a^*/a \to a$, sending one spike to neuron σ_{out}. The spike can pass to neuron σ_{out}. Neuron σ_{out} fires after it receives the spike, and the system halts. The output neuron totally emits one spike to the environment, which indicates that computation result of the logic AND gate is 0 (with input 00).

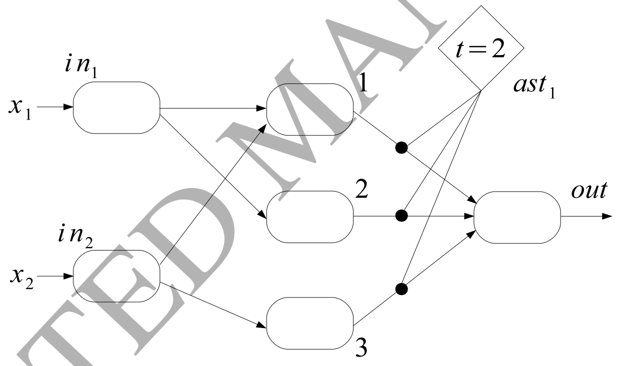

Figure 1: SN P system with astrocyte-like control Π_{AND} simulating a logic AND gate

– If the inputs are $x_1 = 0, x_2 = 1$, neuron σ_{in_1} gets one spike and σ_{in_2} receives two spikes from the environment. Neuron σ_{in_1} fires, resulting with no spike inside and sending one spike to neurons σ_1 and σ_2. Having two spikes inside, neuron σ_{in_2} fires and sends one spike to neurons σ_1 and σ_3 with one spike remaining inside. In this way, neuron σ_1 accumulates two spikes inside, and neurons σ_2 and σ_3 have one spike. Subsequently neurons σ_1, σ_2 and σ_3 fire by using rule $a^*/a \to a$, and each of them sends one spike to neuron σ_{out}. There are three spikes passing along synapses $(1, out)$, $(2, out)$ and $(3, out)$, which are then suppressed by the astrocyte because of the threshold value 2. Meanwhile, neuron σ_{in_2} fires for the second time, sending one spike to each of neurons σ_1 and σ_3. At this moment, neuron σ_1 holds two spikes and neuron σ_3 has one spike. One step later, neurons σ_1 and σ_3 fire for the second time, and send one spike to neuron σ_{out} along synapses $(1, out)$ and $(3, out)$. The two spikes passing along synapses $(1, out)$ and $(3, out)$ are suppressed by astrocyte ast_1. At this moment in time, all the neurons have no spike inside, with the exception that neuron σ_1 holds one spike. It fires one step later, sending one spike to neuron σ_{out}. By receiving the spike, neuron σ_{out} fires sending one spike out. After all the spikes are cleared in all the neurons, the system halts. The output result is the logic AND gate with inputs 0 and 1, which is 0, i.e. one spike.

– If the inputs are $x_1 = 1, x_2 = 0$, neuron σ_{in_1} receives two spikes and σ_{in_2} receives one spike from the environment. The computation process of system Π_{AND} is quite similar to the case of inputs being $x_1 = 0, x_2 = 1$. It is easy to check that in this case, when system Π_{AND} halts, the output neuron totally emits one spike into the environment, which emulates computation result of the logic AND gate being 0.

– If the inputs are $x_1 = 1, x_2 = 1$, each of neurons σ_{in_1} and σ_{in_2} receives two spikes from the environment. In this case, the two neurons can fire twice, sending four spikes to neuron σ_1, and two spikes each to neurons σ_2 and σ_3. The three neurons can fire at the same moment twice. In each time spiking, one spike is consumed in each neuron and three spikes pass along synapses $(1, out)$, $(2, out)$ and $(3, out)$, which will be suppressed by astrocyte ast_1 (with threshold value 2). The last two spikes from neuron σ_1 will be send to neuron σ_{out} one by one in two steps. Eventually neuron σ_{out} receives two spikes from neuron σ_1, and emits two spikes sequentially into the environment, which simulates the computation result of the logic AND gate is 1.

Based on the description of system Π_{AND} above, it is clear that system Π_{AND} can correctly emulate the operation of a logic AND gate. All neurons in system Π_{AND} have the same and unique spiking rule $a^*/a \rightarrow a$, thus system Π_{AND} is simple and homogeneous.

This concludes the proof. □

Theorem 3.2. *The logic OR gate can be emulated by a simple and homogeneous SN P system with astrocyte-like control having ten neurons and three astrocytes.*

Proof An SN P system with astrocyte-like control Π_{OR}, shown in Figure 2, is constructed to emulate the operation of the logic OR gate. The system Π_{OR} is composed of ten neurons with spiking rule $a^*/a \to a$ and three astrocytes. Similarly to what has been done in the previous proof, the four cases 00, 01, 10, 11 are considered.

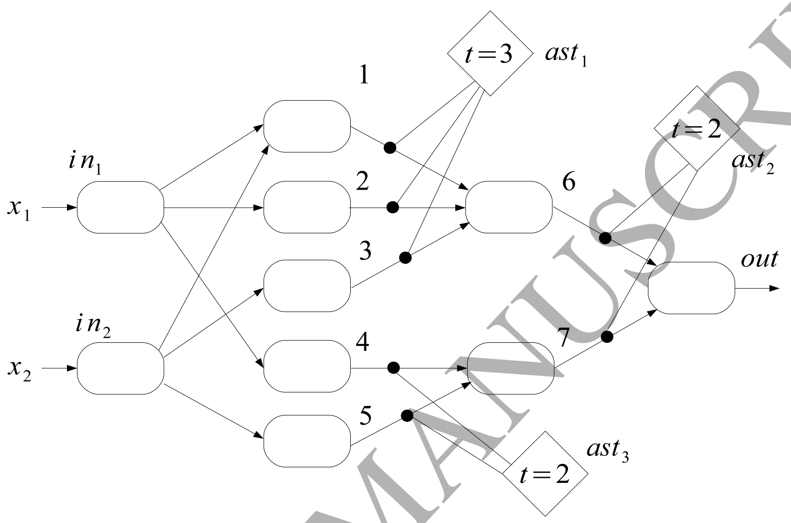

Figure 2: SN P system with astrocyte-like control Π_{OR} simulating a logic OR gate

– If the inputs are $x_1 = 0, x_2 = 0$, each of neurons σ_{in_1} and σ_{in_2} receives one spike from the environment. By using the spiking rule $a^*/a \to a$, neuron σ_{in_1} fires, sending one spike to neurons σ_1, σ_2 and σ_4, meanwhile, neuron σ_{in_2} fires and sends one spike neurons σ_1, σ_3 and σ_5. At this moment, neuron σ_1 accumulates two spikes inside, and each of neurons $\sigma_2, \sigma_3, \sigma_4$ and σ_5 contains one spike. Neurons σ_1, σ_2 and σ_3 fire at the same moment, passing three spikes along synapses $(1, out)$, $(2, out)$ and $(3, out)$. The three spikes will be suppressed by astrocyte ast_1. Neuron σ_1 remains with one spike, and neurons σ_2, σ_3 have no spike left inside. At the same moment, neurons σ_4 and σ_5 fire, sending two spikes along synapses $(4, out)$ and $(5, out)$. The two spikes will be suppressed by astrocyte ast_3 (with threshold value 2), and neurons σ_4, σ_5 hold no spike inside. One step later, neuron σ_1 fires for a second time, sending one spike to neuron σ_6. Then, the number of spikes passing along synapses $(1, 6)$, $(2, 6)$ and $(3, 6)$ is less than 3, so astrocyte ast_1 has excitatory influence on the spike, i.e., the spike can pass to neuron

σ_6. With one spike, neuron σ_6 fires, sending one spike to neuron σ_{out}. By receiving the spike, neuron σ_{out} fires one step later, and the system halts. The output neuron totally emits out one spike to the environment, which means the computation result of the logic OR gate is 0.

– If the inputs are $x_1 = 0, x_2 = 1$, neurons σ_{in_1} and σ_{in_2} receive one spike and two spikes respectively from the environment. Neuron σ_{in_1} fires, consuming the only spike and sending one spike to neurons σ_1, σ_2 and σ_4, meanwhile, neuron σ_{in_2} sends one spike to neurons σ_1, σ_3 and σ_5. One spike is left in neuron σ_{in_2}. One step later, neurons σ_1, σ_2, σ_3, σ_4 and σ_5 fire, but no spike arrives in neurons σ_6 and σ_7 (the spikes are suppressed by astrocytes ast_1 and ast_3). Subsequently, neuron σ_{in_2} fires for the second time emitting one spike, resulting that neuron σ_1 has two spikes, neurons σ_3 and σ_5 have one spike, neurons σ_2 and σ_4 have no spike. Neurons σ_1 and σ_3 fire in the next step, sending two spikes to neuron σ_6. The two spikes can pass to neuron σ_6 due to the the fact that the threshold value of astrocyte ast_1 is 3. One spike remains in neuron σ_1 and neuron σ_3 has no spike inside. By receiving the spike from neuron σ_{in_2}, neuron σ_5 fires with sending one spike to neuron σ_7. Neurons σ_6 and σ_7 fire in the next step, sending two spikes to neuron σ_{out}, but the two spikes are suppressed by astrocyte ast_2. After that, neuron σ_1 fires for a third time by consuming its last spike, sending one spike to neuron σ_6. Then, neuron σ_6 accumulates two spikes, and sends them one by one to neuron σ_{out} by spiking twice. Neuron σ_{out} will fire twice in two steps, and the system finally halts. In total two spikes are emitted into the environment, which indicates computation result of the logic OR gate is 1.

– If the inputs are $x_1 = 1, x_2 = 0$, then neuron σ_{in_1} reads two spikes, and σ_{in_2} gets one spike from the environment. System Π_{OR} works similarly to the case of inputs being $x_1 = 0, x_2 = 1$. It is easy to check that in this case, when system Π_{OR} halts, the output neuron σ_{out} totally emits two spikes into the environment, which indicates the computation result of the logic OR gate is 1.

– If the inputs are $x_1 = 1, x_2 = 1$, each of neurons σ_{in_1} and σ_{in_2} reads two spikes from the environment. In this case, the two neurons can fire twice, sending four spikes to neuron σ_1, and two spikes to each of neurons $\sigma_2, \sigma_3, \sigma_4$ and σ_5. It is easy to check that when neurons σ_1, σ_2 and σ_3 fire, they send three spikes along synapses $(1, 6)$, $(2, 6)$ and $(3, 6)$. The three spikes will be suppressed by astrocyte ast_1. Neurons σ_4 receives two spikes from neuron σ_{Input_1} and neuron σ_5 receives two spikes in total from neuron σ_{in_2} in two steps. The two neurons can fire twice at the same moment, but no spike arrives in neuron σ_7, since the two spikes are suppressed by astrocyte ast_3 with threshold value 2. Neuron σ_1 can fire four times, and it sends one spike to neuron σ_6 in each of the last two spiking. Hence, neuron σ_{out} will receive two spikes and fire twice, emitting two spikes into the environment. After that, the system finally halts, indicating that the computation result of the logic OR gate is 1.

Based on the above description of system Π_{OR}, it is obtained that system Π_{OR} can correctly emulate the operation of a logic OR gate. All neurons in system Π_{OR} have the same and unique spiking rule $a^*/a \to a$, thus system Π_{OR} is simple and homogeneous. Ten neurons and three astrocytes are used in system Π_{OR}.

This concludes the proof. □

Theorem 3.3. *The logic NOT gate can be emulated by a simple and homogeneous SN P system with astrocyte-like control having four neurons and one astrocyte.*

Proof It is constructed an SN P system with astrocyte-like control Π_{NOT} to emulate the operation of logic NOT gate, which is shown in Figure 3. The system Π_{NOT} consists of four neurons and one astrocyte. For the NOT gate, it has the following two cases with input 0 or 1.

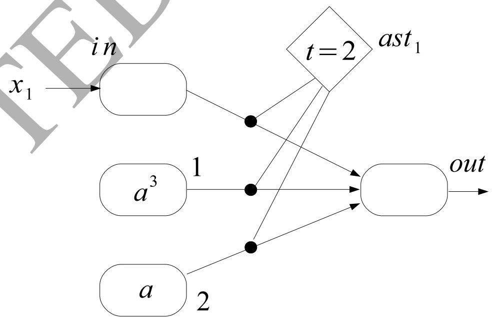

Figure 3: SN P system with astrocyte-like control Π_{NOT} simulating a logic NOT gate

– If the input is $x_1 = 0$, then neuron σ_{in} receives one spike from the environment. Initially, neuron σ_1 has three spikes, and neuron σ_2 has one spike. One step later, neurons σ_{in}, σ_1 and σ_2 fire, sending three spikes along synapses (in, out), $(1, out)$ and $(2, out)$. These spikes are suppressed by astrocyte ast_1. The numbers of spikes in neurons σ_{in} and σ_2 become 0, and the two neurons cannot fire any more. Neuron σ_1 ends with two spikes inside and fires twice in the coming two steps, sending one spike to neuron σ_{out} in each time of spiking. Neuron σ_{out} receives two spikes and will fire twice in two steps. When system Π_{NOT} halts, neuron σ_{out} has emitted two spikes into the environment totally, which simulates the simulation result of the logic NOT gate is 1.

- If the input is $x_1 = 1$, neuron σ_{in} gets two spikes from the environment. Neuron σ_1 initially has three spikes, and neuron σ_2 has one spike. One step later, neurons σ_{in}, σ_1 and σ_2 fire, sending three spikes along synapses (in, out), $(1, out)$ and $(2, out)$. These spikes are suppressed by astrocyte ast_1 with threshold value 2. After that, neurons σ_{in} and σ_1 fire again, sending two spikes along synapses (in, out), $(1, out)$. The two spikes are suppressed by astrocyte ast_1, due to the fact that the threshold value of astrocyte ast_1 is 2. At that moment, all the neurons contain no spike, except for neuron σ_1 having one spike inside. With one spike, neuron σ_1 fires one step later, sending one spike to neuron σ_{out}. Neuron σ_{out} emits the spike into the environment and system Π_{NOT} halts, indicating the computation result of the logic NOT gate is 0.

From the above description it is found that system Π_{NOT} can correctly simulating the operation of a logic NOT gate. There are four neurons in system Π_{NOT}, which have the same and unique spiking rule, thus system Π_{NOT} is simple and homogeneous.

This concludes the proof. □

Theorem 3.4. *The logic NOR gate can be emulated by a simple and homogeneous SN P system with astrocyte-like control having four neurons and one astrocyte.*

Proof An SN P system with astrocyte-like Π_{NOR} is constructed to emulate the operation of a logic NOR gate. System Π_{NOR} is shown in Figure 4, and is composed of four neurons and one astrocyte. It has the following four cases with input digits 00, 01, 10 and 11, respectively.

- If the inputs are $x_1 = 0, x_2 = 0$, each of neurons σ_{in_1} and σ_{in_2} receives one spike from the environment, while neuron σ_1 has 3 spikes initially. The three neurons fire, sending three spikes to neuron σ_{out}, which are suppressed

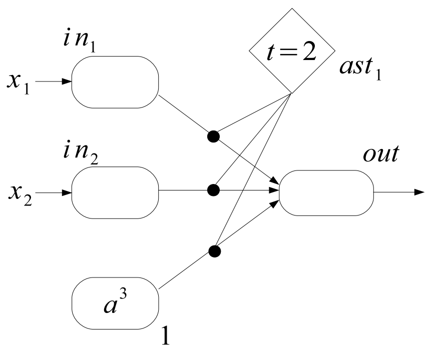

Figure 4: SN P system with astrocyte-like control Π_{NOR} simulating a logic NOR gate

by astrocyte ast_1. After that, neuron σ_1 ends with two spikes and will fire twice in the coming two steps. At each time of spiking, it sends one spike to neuron σ_{out}. In this way, neuron σ_{out} receives two spikes, each in a step and can fire for twice. When system Π_{NOR} halts, neuron σ_{out} emits in total two spikes into the environment, which indicates the computation result of the logic NOR gate is 1.

– If the inputs are $x_1 = 0, x_2 = 1$, neuron σ_{in_1} receives one spike, and neuron σ_{in_2} receives two spikes from the environment. Three spikes are initially placed in neuron σ_1. In the next step, the three neurons fire at the same moment, sending three spikes to neuron σ_{out}. The three spikes are suppressed by astrocyte ast_1 (with threshold value 2). After that, neuron σ_1 has two spikes, and neuron σ_{in_2} has one spike. The two neurons fire one step later, sending two spikes to neuron σ_{out}. The two spikes are also suppressed by astrocyte ast_1. At this moment of time, only neuron σ_1 ends with one spike and fires for a third time. Then, it sends one spike to neuron σ_{out}. By receiving one spike, neuron σ_{out} emits one spike in total into the environment, which indicates that the computation result of the logic NOR gate is 0.

  Similarly, it is easy to prove that when the input digits are $x_1 = 1, x_2 = 0$, neuron σ_{out} emits in total one spike into the environment, indicating that the computation result of the logic NOR gate is 0.

– If the inputs are $x_1 = 1, x_2 = 1$, each of neurons σ_{in_1} and σ_{in_2} receives two spikes from the environment, and neuron σ_1 has three spikes initially. In this case, neurons σ_{in_1}, σ_{in_2} and σ_1 fire at the same moment twice. At each time of spiking, they send three spikes to neuron σ_{out}, which are then suppressed by astrocyte ast_1. After that, only neuron σ_1 has one spike. It fires and emits one spike to neuron σ_{out}. By receiving the spike, neuron σ_{out} fires and emits one spike into the environment, and the system halts. This indicates that the computation result of the logic NOR gate is 0.

From the above description, it is proved that system Π_{NOR} can correctly emulate the operation of a logic NOR gate. There are four neurons and one astrocyte in system Π_{NOR}. All neurons in system Π_{NOR} have the same and unique spiking rule, thus system Π_{NOR} is simple and homogeneous.

This concludes the proof. □

From system Π_{NOT} illustrated in Figure 3 and system Π_{NOR} shown in Figure 4, it is found that system Π_{NOT} is specific case in system Π_{NOR} with initially placing one spike (encoding input 0) in neuron σ_{in_2}. This is due to the fact that $NOT(x) = NOR(x, 0)$, i.e., NOT gate for input x has the same behaviour as NOR gate for inputs x and 0.

Theorem 3.5. *The logic XOR gate can be emulated by a simple and homogeneous SN P system with astrocyte-like control having seven neurons and three astrocytes.*

Proof We construct an SN P system with astrocyte-like control Π_{XOR} to emulate the operation of a logic XOR gate. The system, shown in Figure 5, consists of seven neurons and three astrocytes. In the following, we explain the work of the system with input digits 00, 01, 10 and 11, respectively. Initially, neuron σ_1 has two spikes and σ_2 has three spikes.

– If the inputs are $x_1 = 0, x_2 = 0$, each of neurons σ_{in_1} and σ_{in_2} receives one spike from the environment. Neurons σ_{in_1} and σ_1 fire at the same moment, sending two spikes to neuron σ_3. The two spikes are suppressed by astrocyte ast_2, and neuron σ_1 ends with one spike inside. Meanwhile, neurons σ_{in_2} and σ_2 fire, sending two spikes to neuron σ_4. The two spikes are suppressed by astrocyte ast_1. After that, neuron σ_2 has two spikes inside. One step later, neurons σ_1 fires with sending one spike to neuron σ_3, meanwhile, neuron σ_2 sends one spike to neuron σ_4. Having one spike inside, neurons σ_3 and σ_4 fire, sending two spikes to neuron σ_{out}. The two spikes cannot arrive to neuron σ_{out} due to the inhibitory function of astrocyte ast_3. Neuron σ_2 fires for the third time, emitting one spike to neuron σ_4, and the spike will be sent to neuron σ_{out} from neuron σ_4. In this way, neuron σ_{out} emits one spike into the environment in total and the system halts, indicating that the computation result of the logic XOR gate is 0.

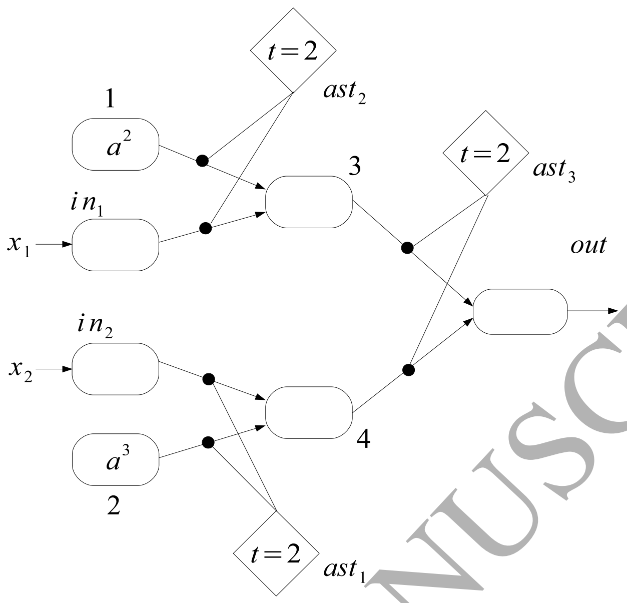

Figure 5: SN P system with astrocyte-like control Π_{XOR} simulating a logic XOR gate

- If the inputs are $x_1 = 0, x_2 = 1$, neuron σ_{in_1} receives one spike, and neuron σ_{in_2} receives two spikes from the environment. Neurons σ_{in_1} and σ_1 fire, sending two spikes to neuron σ_3. The two spikes are suppressed by astrocyte ast_2. One step later, neuron σ_1 fires and sends one spike to neuron σ_3. Neuron σ_3 can fire and fire the spike to neuron σ_{out}. By receiving the spike, neuron σ_{out} emits one spike into the environment. Neurons σ_{in_2} and σ_2 fire twice. In each time of spiking, they send two spikes to neuron σ_3, and the two spikes are suppressed by astrocyte ast_1. Neuron σ_2 ends with one spike, and fires sending one spike to neuron σ_4. Having one spike inside, neuron σ_4 sends one spike to neuron σ_{out}. By receiving the spike, neuron σ_{out} fires, emitting the second spike into the environment, and the system halts. In this case, neuron σ_{out} emits two spikes in total into the environment, indicating that the computation result of the logic XOR gate is 1.

- If the inputs are $x_1 = 1, x_2 = 0$, neuron σ_{in_1} receives two spikes, and neuron σ_{in_2} receives one spike from the environment. In this case, neuron σ_{in_1} and σ_1 will fire twice, and no spike arrives in neuron σ_3 due to the inhibitory function of astrocyte ast_2. Neuron σ_{in_2} has one spike, and neuron σ_2 has three spikes. When neurons σ_{in_2} and σ_2 fire, they send two spikes along synapses $(in_2, 4)$ and $(2, 4)$. The two spikes are suppressed by astrocyte ast_1 (with threshold value 2). After that, neuron σ_2 has two spikes left within, thus fires twice, sending two spikes to neuron σ_4. In this way, neuron σ_{out} emits two spikes in total into the environment, indicating that the computation result of the logic XOR gate is 1.

– If the inputs are $x_1 = 1, x_2 = 1$, each of neurons σ_{in_1} and σ_{in_2} receives two spikes from the environment. In this case, neuron σ_{in_1} and σ_1 will fire twice in the coming two steps, and no spike can arrive at neuron σ_3 due to the inhibitory function of astrocyte ast_2. At each time of spiking, neurons σ_{in_2} and σ_2 send two spikes along synapses $(in_2, 4)$ and $(2, 4)$, but the spikes are suppressed by astrocyte ast_1 (with threshold 2). After that, neuron σ_2 has one spike, and sends the spike to neuron σ_4. In this case, neuron σ_{out} receives one spike, and emits the spike into the environment, indicating that the computation result of the logic XOR gate is 0.

From the above description, it is proved that system Π_{XOR} can correctly emulate the operation of logic NOR gate. There are seven neurons and three astrocytes in system Π_{XOR}. All neurons in system Π_{XOR} have the same and unique spiking rule, thus system Π_{XOR} is simple and homogeneous.

This concludes the proof. □

Theorem 3.6. *The logic NAND gate can be emulated by a simple and homogeneous SN P system with astrocyte-like control having eight neurons and one astrocyte.*

Proof An SN P system with astrocyte-like Π_{NAND}, shown in Figure 6, is constructed to emulate the operation of logic NAND gate. System Π_{NAND} is composed of eight neurons and one astrocyte. It has the following four cases with input digits 00, 01, 10 and 11, respectively.

– If the inputs are $x_1 = 0, x_2 = 0$, each of neurons σ_{in_1} and σ_{in_2} receives one spike from the environment. Neurons σ_{in_1} and σ_{in_2} fire with sending two spikes to neuron σ_3. Meanwhile, neuron σ_1 sends one spike to neuron σ_4, and neuron σ_2 sends one spike to neuron σ_5. In the next step, neurons σ_3, σ_4 and σ_5 fire to send three spikes to neuron σ_{out}, and the three spikes are suppressed by astrocyte ast_1 (with threshold value 2). Neuron σ_1 initially has five spikes, so it can fire for five times, sending in total five spikes to neuron σ_4. Neuron σ_2 will fire for three times and sends one spike to neuron

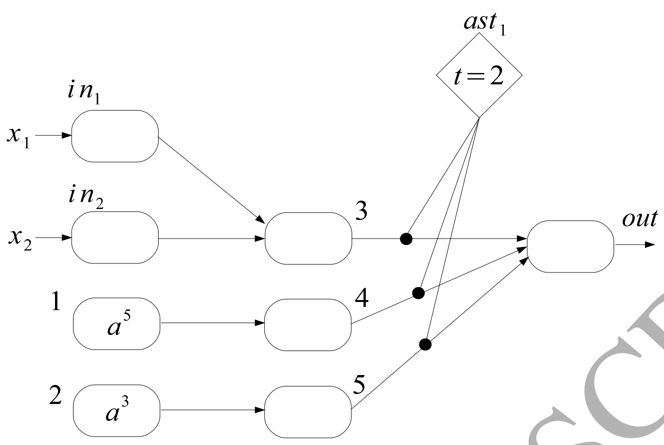

Figure 6: SN P system with astrocyte-like control Π_{NAND} simulating a logic NAND gate

σ_5 in each time spiking. It is not hard to find that only when the numbers of spikes in both of neurons σ_3 and σ_5 become 0, the spike from neuron σ_4 can pass to neuron σ_{out}. When neurons σ_3 and σ_5 have no spike inside, neuron σ_4 remains with two spikes. The two spikes can pass to neuron σ_{out} one by one, and will be emitted into the environment from neuron σ_{out}, indicating that the computation result of the logic NAND gate is 1 with input 00.

– If the inputs are $x_1 = 1, x_2 = 0$, neuron σ_{in_1} receives two spikes, and neuron σ_{in_2} receives one spike from the environment. Neuron σ_{in_1} can fire twice, sending two spikes in two steps to neuron σ_3, and neuron σ_{in_2} emits one spike to neuron σ_3. Having five spikes inside, neuron σ_1 will fire for five times, sending in total five spikes to neuron σ_4, while neuron σ_2 sends three spikes to neuron σ_5. In this case, when neurons σ_3 and σ_5 have no spike inside, the spike from neuron σ_4 can pass to neuron σ_{out}. By receiving two spikes from neuron σ_4, neuron σ_{out} fires twice, sending in total two spikes into the environment. It means the computation result of the logic NAND gate is 1.

– If the inputs are $x_1 = 0, x_2 = 1$, neuron σ_{in_1} receives one spike, and neuron σ_{in_1} receives two spikes from the environment. In this case, neuron σ_{in_1} emits one spike to neuron σ_3, while neuron σ_{in_2} fires twice, sending two spikes to neuron σ_3 in two steps. Similar to the case with input digit 10, when neurons σ_3 and σ_5 have no spike inside, the spike from neuron σ_4 can pass to neuron σ_{out}. Neuron σ_{out} receives two spikes from neuron σ_4

and fires twice, emitting two spikes into the environment. This indicates the computation result of logic NAND gate is 1.

– If the inputs are $x_1 = 1, x_2 = 1$, each of neurons σ_{in_1} and σ_{in_2} receives two spikes from the environment, both of neurons σ_{in_1} and σ_{in_2} can fire twice, thus sending in total 4 spikes to neuron σ_3. In this case, when neurons σ_3 and σ_5 have no spike inside, neuron σ_4 remains with one spike. Neuron σ_{out} gets one spike from neuron σ_4 and emits the spike into the environment. indicating that the computation result of logic NAND gate is 0.

From the above description, it is proved that system Π_{NAND} can correctly emulate the operation of a logic NAND gate. There are eight neurons and one astrocyte in system Π_{NAND}. All neurons in system Π_{NAND} have the same and unique spiking rule, thus system Π_{NAND} is simple and homogeneous. □

Mathematically, the results introduced in this section can be summarized by the following form.

Let $X \in \{AND, OR, NOT, NOR, XOR, NAND\}$. There exists an SN P system Π and an encoding e such that for every $p, q \in \{0, 1\}$ $\Pi(e(p), e(q)) = X(p, q)$, where Π be an SN P systems with homogenous neurons and astrocytes-like control. The encoding e encodes number 0 by one spike and number 1 by two spikes. Note that, $X(p, q)$ is the truth value of pXq, for every logic gate X, and $\Pi(e(p), e(q))$ is the output of system Π on input $e(p)$ and $e(q)$.

4. An Example of Neural-like Boolean Circuit

With the neural-like logic gates designed in Section 3, an example of neural-like Boolean circuit is developed by connecting neural-like logic gates. Since different logic gates may spend different steps to complete their computations, it needs synchronization modules to regulate the outputs of different logics gates to enter the other logic gate as inputs synchronously. The basic function of the synchronization module is to make certain steps of delay for spikes entering into certain neurons. The synchronization module for making n steps delay is shown in Figure 7.

It is not hard to find that when a spike enters into neuron σ_1, it takes n steps to pass to neuron σ_n, and at step $n + 1$ the spike passes out of neuron σ_n. The Boolean circuit shown in Figure 8 from [18] is reconstructed here based on logic gates by SN P systems proposed.

In the following, the logic gates constructed in Section 3 are applied to design Boolean circuit in Figure 8. It can be easily checked that the homogenous SN

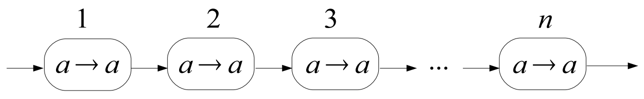

Figure 7: The synchronization module for making n steps delay

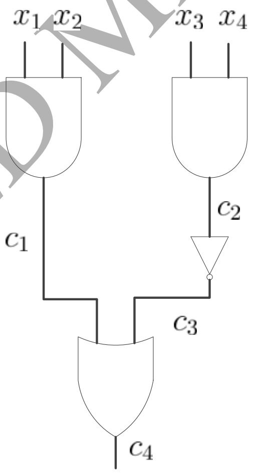

Figure 8: An example of Boolean circuit from [18]

P system with simple neurons and asytrocyte-like control constructed in Figure 9 can mimic the computation of Boolean circuit from Figure 8. The details are omitted here.

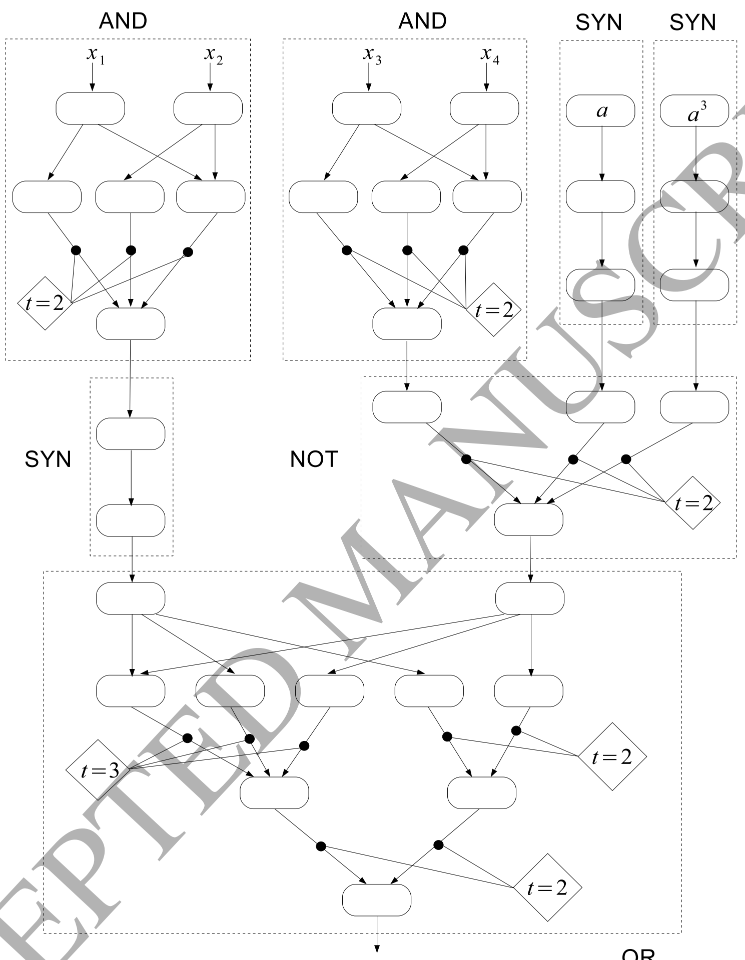

Figure 9: The homogenous SN P system with simple neurons and asytrocyte-like control to simulate the Boolean circuit in Figure 8, where SYN denotes a synchronous module.

5. Conclusion

In this research, we proposed and established SN P systems with astrocyte-like control to emulate logic AND, OR, NOT, NOR, XOR and NAND gates. The obtained SN P systems are simple and homogeneous, which means that each neuron in the systems has the same and unique spiking rule. A Boolean circuit is built by the SN P systems for logic gates, where synchronization modules are designed to regulate the outputs of different logics gates to enter the other logic gate as inputs synchronously. These results demonstrate a novel way of constructing "neural-like" logic gates and circuits with one type of universal neurons, whose function is fairly simple, like a "transmitter of information". With these neural-like logic gates, Boolean circuits with cascade connections of logic gates can be constructed, thus can mimic finite computing devices, such as finite transducers. This work sheds some light on potential directions to theoretically design neural circuits. The results have some merit interpretation: the individual neuron is simple in structure and universal, while with the modulation of astrocytes and cooperating with each other, a network of neurons and astrocytes can be rather powerful. With these neural-like logic gates, Boolean circuits with cascade connections of logic gates can be constructed, thus can mimic finite computing devices, such as fine transducers. Since the logic gates cannot return to its initial configuration after logic computation, some extra auxiliary synchronization modules might be needed to "fill" spikes for the gates.

The numbers of neurons in the systems constructed is not optimized. Further research to reduce the numbers of neurons used in SN P systems with astrocyte-like control simulating logic gates is attemptable. Also, it will be of great interest to simplify the topological structures of the systems. A possible approach is to use synchronous, asynchronous and local synchronous strategies to regulate neuron spiking. In our systems, the binary number 0 is encoded by one spike, and the binary number 1 is represented with two spikes. Some novel efficient information encoding strategies, such as introducing anti-spikes in the systems, can also be looked into.

For the systems constructed in this work, they cannot return to the initial configuration when halting. That is a weakness that result in the systems cannot be reused. It is an open problem to design reused logic gates with simple and homogeneous neurons by SN P systems with astrocyte-like control. There might be two possible ways that deserve further research. (1) Adding "spikes supplement" modules to supply particular numbers of spikes to certain neurons to return the systems to their initial configurations when the systems halt. (2) Redesigning the systems without placing any spikes in the neurons initially. It is known that SN P systems are robust to the spiking time, see time-free SN P systems [22, 28]. It is worthy to design time-robust logic gates with time-free SN P systems.

To emulate the logic gates, an SN P system with astrocyte-like control costs multiple computation steps to complete its simulation. How to speed up the simulations is worthy for further research. For complex logic gates, with three inputs for instance, would it be possible to construct simple and homogeneous SN P systems with astrocyte-like control to emulate them? The systems proposed in this work provide theoretical models to construct neural-like logic gates with universal information processing units. It will be useful if parallel hardware, such as GPU, can be utilized to realize the neural-like digital logic gates and logic circuits.

As a new candidate in spiking neural networks, the learning strategies for SN P systems is of great research interests. A possible way is to borrow some learning ideas from the artificial neural networks [10, 13, 14], Deep learning strategy [9, 36] and machine learning algorithms [4, 8, 51, 56].

Acknowledgment

The research is under the auspices of National Natural Science Foundation of China (No. 61402187, 61572522, 61572523 and 61502535), China Postdoctoral Science Foundation funded project (2016M592267), and Fundamental Research Funds for the Central Universities (R1607005A). One of the researchers of this work, Tao Song, is under the sponsorship of the visiting fellowship scheme of Swinburne University of Technology Sarawak Campus.

References

[1] P. Bezzi, G. Carmignoto, L. Pasti, Prostaglandins stimulate calcium-dependent glutamate release in astrocytes, Nature 391 (6664) (1998) 281–285.

[2] A. Binder, R. Freund, M. Oswald, L. Vock. Extended spiking neural P systems with excitatory and inhibitory astrocytes, Proceedings of the 8th WSEAS International Conference on Evolutionary Computing, Vancouver, British Columbia, Canada, June 19-21, 2007

[3] F.G.C. Cabarle, H.N. Adorna, M.J. Perez-Jimenez, T. Song. Spiking neural P systems with structural plasticity. Neural Computing and Applications. 26 (2015) 1905-1917.

[4] F. Campuzano, T. Garcia-Valverde, J.A. Botia, E. Serrano, Generation of human computational models with machine learning, Information Sciences, 293 (2015) 97–114.

[5] M. Cavaliere, O. H. Ibarra, G. Păun, O. Egecioglu, M. Ionescu, S. Woodworth, Asynchronous spiking neural P systems, Theoretical Computer Science 410 (24) (2009) 2352–2364.

[6] R. Ceterchi, D. Sburlan, Simulating Boolean circuits with P systems, Lecture Notes in Computer Science, 2933 (2004) 104-122.

[7] Z. Chen, P. Zhang, X. Wang, X. Shi, T. Wu, P. Zheng, A computational approach for nuclear export signals identification using spiking neural P systems, Neural Computing and Applications, (2016) DOI: 10.1007/s00521-016-2489-z.

[8] I. Couso, L. Sanchez, Machine learning models, epistemic set-valued data and generalized loss functions: An encompassing approach, Information Sciences, 358 (2016) 129-150.

[9] D. Erhan, Y. Bengio, A. Courville, Why does unsupervised pre-training help deep learning, Journal of Machine Learning Research 11 (2010) 625–660.

[10] R. Gao, Y. Wang, J. Lai, Neuro-Adaptive Fault-tolerant control of high speed trains under traction-braking failures using self-structuring neural networks, Information Sciences, 367–368 (2016) 449–462.

[11] M. Gheorghe, S. Konur, F. Ipate, Kernel P systems and stochastic P systems for modelling and formal verification of genetic logic gates (2016), in press.

[12] A. M. Gutierrez-Naranjo, A. Leporati, First steps towards a CPU made of spiking neural P systems. Int. J. of Computers, Communications & Control, Vol. IV(3) (2009) 244-252.

[13] S. Haykin, Neural networks and learning machines, Upper Saddle River, New Jersy, USA Pearson, 2009.

[14] R. He, J. Tang, P. Gong P, Multi-document summarization via group sparse learning, Information Sciences, 349 (2016), 12–24.

[15] O. H. Ibarra, A. Păun, A. Rodríguez-Patón, Sequential SNP systems based on min/max spike number, Theoretical Computer Science 410 (30) (2009) 2982–2991.

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

[17] M. Ionescu, T.O. Ishdorj, Boolean circuits and a DNA algorithm in membrane computing, Lecture Notes in Computer Science 3850 (2006) 272–291.

[18] M. Ionescu, D. Sburlan, Several applications of spiking neural P systems. Computing and Informatics, 27 (2008) 515–528

[19] N. Kasabov, E. Capecci, Spiking neural network methodology for modelling, classification and understanding of EEG spatio-temporal data measuring cognitive processes, Information Sciences, 294 (2015) 565–575.

[20] K. Krithivasan, V. P. Metta, D. Garg, On string languages generated by spiking neural P systems with anti-spikes, International Journal of Foundations of Computer Science 22 (01) (2011) 15–27.

[21] X. Li, Z. Wang, W. Lu, A spiking neural system Based on DNA strand displacement. Journal of Computational and Theoretical Nanoscience, 2015, DOI: 10.1166/jctn.2015.3732

[22] X. Liu, Z. Li, J. Liu, L. Liu, X. Zeng, Implementation of arithmetic operations with time-free spiking neural P systems, IEEE Trans Nanobioscience, 14 (6) (2015) 617–624.

[23] W. Maass, Networks of spiking neurons: the third generation of neural network models, Neural Networks 10 (9) (1997) 1659–1671.

[24] W. Maass, C. M. Bishop, Pulsed neural networks, MIT Press, 2001.

[25] L.F. Macias-Ramosb, L. Pan, M.J. Perez-Jimenez, Time-free solution to sat problem using P systems with active membranes, Theoretical Computer Science, 529 (2014) 61–68.

[26] V.P. Metta, K. Krithivasan, D. Garg, Some characteristics of spiking neural P systems with anti-spikes, In Proceedings of the 11th International Conference on Membrane Computing, Jena, Germany, (2010) 291–303.

[27] L. Pan, Gh. Păun, Spiking neural P systems with anti-spikes, Int. J. of Computers, Communications & Control, IV (3) (2009) 273–282.

[28] L. Pan, X. Zeng, X. Zhang. Time-free spiking neural P systems. Neural Computation, 23 (5) (2011) 1320–1342.

[29] Gh. Păun, Membrane computing: an introduction, Springer, 2002.

[30] Gh. Păun, M. J. Pérez-Jiménez, Spiking neural P systems: an overview, in: W. B. A.B. Porto, A. Pazos (Ed.), Advancing Artificial Intelligence through Biological Process Applications, PA: Medical Information Science Reference, Hershey, 2008, pp. 60–73.

[31] Gh. Păun, G. Rozenberg, A. Salomaa, The Oxford handbook of membrane computing, Oxford University Press, 2010.

[32] Gh. Păun, Spiking neural P systems with astrocyte-like control, Journal of Universal Computer Science 13 (11) (2007) 1707–1721.

[33] H. Peng, J. Wang, M. J. Pérez-Jiménez, H. Wang, J. Shao, T. Wang, Fuzzy reasoning spiking neural P system for fault diagnosis, Information Sciences 235 (2013) 106–116.

[34] G. Perea, M. Navarrete, A. Araque, Tripartite synapses: astrocytes process and control synaptic information, Trends in Neurosciences 32 (8) (2009) 421–431.

[35] G. Rozenberg, A. Salomaa, Handbook of Formal Languages, Vol. 3, Springer-Verlag, Berlin, 1997.

[36] J. Schmidhuber. Deep learning in neural networks: an overview, Neural Networks, 61 (2015) 85-117.

[37] T. Song, L. Pan, K. Jiang, B. Song, W. Chen, Normal forms for some classes of sequential spiking neural P systems, IEEE Trans. on Nanobioscience 12 (3) (2013) 255–264.

[38] T. Song, L. Pan, Gh. Păun, Spiking neural P systems with rules on synapses, Theoretical Computer Science 529 (2014) 82–95.

[39] T. Song, L. Pan, Spiking neural P systems with rules on synapses working in maximum spikes consumption strategy, IEEE Trans. on Nanobioscience 14 (1) (2015) 38–44.

[40] T. Song, L. Pan, Spiking neural P systems with rules on synapses working in maximum spiking strategy, IEEE Trans. on Nanobioscience 14 (4) (2015) 465–477.

[41] T. Song, L. Pan, Spiking neural P systems with request rules, Neurocomputing, 193 (2016) 193–200

[42] T. Song, L. Pan, Gh. Păun, Asynchronous spiking neural P systems with local synchronization, Information Sciences 219 (2012) 197–207.

[43] X. Shi, Z. Wang, C. Deng, T. Song, L. Pan, Z. Chen, A novel bio-sensor based on DNA strand displacement, PloS ONE, 9 (10) (2015) e108856.

[44] X. Shi, X. Wu, T. Song, X. Li, Construction of DNA nanotubes with controllable diameters and patterns by using hierarchical DNA sub-tiles, Nanoscale, (2016) DOI: 10.1039/C6NR02695H

[45] J. Wang, P. Shi, H. Peng, M. J. Pérez-Jiménez, T. Wang, Weighted fuzzy spiking neural P systems, IEEE Trans. on Fuzzy Systems, 21 (2) (2013) 209–220.

[46] J. Wang, H. Peng, Adaptive fuzzy spiking neural P systems for fuzzy inference and learning, International Journal of Computer Mathematics 90 (4) (2013) 857–868.

[47] T. Wang, G. Zhang, M. J. Pérez-Jiménez, Fuzzy membrane computing: theory and applications. International Journal of Computers, Communications & Control 10 (6) (2015) 904-935.

[48] T. Wang, G. Zhang, J. Zhao, Z. He, J. Wang, M. J. Pérez-Jiménez, Fault diagnosis of electric power systems based on fuzzy reasoning spiking neural P systems, IEEE Trans. on Power System, 30 (3) (2015) 1182–1194.

[49] X. Wang, T. Song, F. Gong, P. Zheng, On the computational Power of Spiking neural P systems with self-orgniaztion, Scientific Reports, 6 (2016) Article number: 27624.

[50] T. Wu, Z. Zhang, Gh. Păun, Cell-like spiking neural P systems. Theoretical Computer Science, 623 (2016) 180–189.

[51] Z. Xie, J. Sun, V. Palade, Evolutionary sampling: a novel way of machine learning within a probabilistic framework, Information Sciences, 299 (2015), 262–282.

[52] G. Zhang, H. Rong, F. Neri, M. J. Pérez-Jiménez, An optimization spiking neural P system for approximately solving combinatorial optimization problems, International Journal of Neural Systems 24 (05) Article ID: 1440006.

[53] X. Zhang, X. Zeng, B. Luo, L. Pan, On some classes of sequential spiking neural P systems, Neural Computation 26 (5) (2014) 974–997.

[54] J. Zhang, Y. Zhu, Y. Pan Y, Efficient parallel boolean matrix based algorithms for computing composite rough set approximations, Information Sciences, 329 (2016) 287–302.

[55] L. Zhang, P.N. Suganthan. A survey of randomized algorithms for training neural networks, Information Sciences, (2016) DOI: 10.1016/j.ins.2016.01.039.

[56] C. Zhu, Z. Wang, D. Gao, Double-fold localized multiple matrixized learning machine, Information Sciences 295 (2015) 196–220.