Table of Contents

Taking the Pulse of SN P Systems: a Quick Survey

Henry N. Adorna[1], Francis George C. Cabarle[1], Luis F. Macías-Ramos[2], Linqiang Pan[3], Mario J. Pérez-Jiménez[2], Bosheng Song[3], Tao Song[3,4], and Luis Valencia-Cabrera[2]

[1] Algorithm & Complexity Lab, Department of Computer Science, University of the Philippines Diliman, Quezon City, the Philippines
[2] Research Group on Natural Computing, Department of Computer Science and Artificial Intelligence, University of Seville, Avda. Reina Mercedes s/n, 41012 Sevilla, Spain
[3] Key Laboratory of Image Information Processing and Intelligent Control, School of Automation, Huazhong University of Science and Technology, Wuhan 430074, Hubei, China,
[4] Faculty of Engineering, Computing and Science, Swinburne University of Technology Sarawak Campus, Kuching 93350, Malaysia

Abstract. Spiking neural P systems comprise one of the main branches of membrane computing. These systems take inspiration from the structure and behaviour of neuronal cells within the nervous system and have proven their ability to solve computationally hard problems in polynomial time. The related models have become very popular in recent years, with new variants appearing in a fast succession. Keeping on track with this development is difficult, specially for newcomers. As such, this chapter deals with a selection of variants of this kind of P systems, intended to address a fistful of variants, focusing on their diversity. Special attention should be given to fuzzy spiking neural P systems - an extensive survey can be found in [24].

Keywords: Membrane computing, P system, Spiking neural P system, Asynchronous SN P system, Astrocyte, Anti-spike, SN P system with plasticity

1 Introduction

Spiking neural P systems (SN P systems, for short) are a variant of P systems, corresponding to a shift from *cell-like* to *neural-like* architectures. They are formally introduced in [12], and take inspiration from the way in which neurons in the brain exchange information by means of the propagation through their synapses of electrical impulses called action potentials or spikes.

Neurons are specialized cells characterized by their electrical excitability and the presence of synapses. Synapses are complex membrane junctions that interconnect neurons allowing transmission of neuronal signals among them.

A neural signal is an electric pulse called action potential or spike. Neuronal signals are typically identical, with the form of the pulse not changing as the action potential propagates. So, neuronal signals themselves do not carry any information. Instead, the information is encoded in the number and timing of spikes. According to the amount of pulses received by a neuron during a period of time, such pulses may have an inhibitory or excitatory effect over the neuron. In the first case, the neuron does not generate any pulse in response to its input spikes. In the excitatory case, the neuron generates a pulse, which originates at the soma and propagates rapidly along the axon, activating synapses onto other neurons as it goes by. There exists a refractory period associated to neurons: even with very strong input, it is impossible to excite a second spike during or immediately after a first one. The minimal (time) distance between two spikes defines the refractory period of the neuron. Working as described above, a neuron generates impulses at regular or irregular intervals, giving place to a sequence of action potentials called spike train.

SN P systems incorporate the key elements of the structure and functioning of neurons and synapses described above. In SN P systems, cells, called *neurons* in this case, are placed in the nodes of a directed graph, called the *synapse graph*. The content of any neuron consists of a number of copies of a single object type, called *spike*. Every neuron may also have associated a number of *firing* and *forgetting* rules. Firing rules allow a neuron to send information to its neighbours by means of spikes, which are accumulated at target neurons. Executing a firing rule involves removing a certain amount of spikes from the neuron and emitting a spike that is replicated along the outgoing synapses and finally stored at target neurons. This is accomplished in a two-stage process: firstly the spikes are removed from the neuron, and after a specific period of time, which depends on the rule, the neuron fires the output spike. During this period, the neuron becomes "closed" (inactive): it does not accept new spikes and cannot "fire" any (firing or forgetting) rule. The period of time is specified by a delay parameter associated with the firing rule. On the other hand, forgetting rules simply remove a certain amount of spikes from the neuron, with no spike being emitted. The applicability of a rule is determined by checking the neuron content against a regular expression associated with the rule. If more than one rule is applicable, then one of them is non-deterministically chosen. As usually happens in membrane systems, a global clock is assumed, which marks the evolution of the system, thus making it work in a synchronized way. While individual neurons works sequentially (at most one rule can be executed at any time by a neuron), the system as a whole works in parallel, since different neurons can execute rules simultaneously.

It is easy to see that the above model captures, in a general way, the structure and functioning of neurons and synapses. Since in real neurons the shape and size of pulses is not important, it is enough to have a single type of object, the spike. Following this, the inhibitory or excitatory effect of the received impulses over a neuron is determined by the number of spikes received over time by such neuron so that in the model, a neuron accumulates spikes to count the number of received impulses. When the amount of spikes within the neuron reaches certain levels, firing or forgetting rules become applicable. Executing a firing rule corresponds to the case in which the impulses have an excitatory influence over the neuron: a certain amount of spikes is removed from the neuron and a single spike is sent along the outgoing synapses, which models the synapse activation process. Executing a forgetting rule corresponds, in turn, with the inhibitory scenario: spikes are removed from the neuron and no spike is sent out.

SN P systems specified above were originally introduced in [12], collectively known as *classic* SN P system models. Since their introduction, many variants of SN P systems have been developed and their computational properties were studied. The remainder of this chapter is structured as follows. Firstly, the formal definition of the classic SN P systems is reviewed. Next, some interesting variants are presented, dealing with asynchronous SN P systems, astrocytes, anti-spikes and SN P systems with plasticity. Finally, a brief summary of useful references dealing with theoretical results is included.

2 Classical Spiking Neural P Systems

The definition of classic spiking neural P systems [12] is as follows.

Definition 1. *An SN P system of degree $m \geq 1$ is a construct of the form:*

$$\Pi = (O, \sigma_1, \sigma_2, \ldots, \sigma_m, syn, in, out),$$

where:

1. *$O = \{a\}$ is the singleton alphabet (a denotes* spike*);*
2. *$\sigma_1, \sigma_2, \ldots, \sigma_m$ are neurons of the form $\sigma_i = (n_i, R_i)$ where:*
   - *$n_i \geq 0$ is the initial number of spikes contained in σ_i;*
   - *R_i is a finite set of rules of the two following forms:*
     (1) $E/a^c \to a; d$, with E a regular expression over O, $c \geq 1, d \geq 0$;
     (2) $a^s \to \lambda$, with $s \geq 1$ and the restriction $a^s \notin L(E)$ for any rule of type (1) $E/a^c \to a; d \in R_i$;
3. *$syn \subseteq \{1, 2, \ldots, m\} \times \{1, 2, \ldots, m\}$, with $(i, i) \notin syn$ for $1 \leq i \leq n$ is the synapse graph, which defines synapses among neurons;*
4. *$in, out \in \{1, 2, \ldots, m\}$ are the input and output neurons respectively;*

The rules of type (1), with the form $E/a^c \to a; d$, are called *standard firing* (or *spiking*) rules. The term standard refers to the fact that only one copy of spike appears in the right hand side of the rule, which corresponds to the biological interpretation in the last section. For simplicity, we will refer to them simply as firing rules. As mentioned above, E is a regular expression over O. When $L(E) = a^c$, the rule can be written as $a^c \to a; d$. On the other hand, d is a non-negative integer known as *delay*. When $d = 0$, the rule can be written as $E/a^c \to a$. If both $L(E) = a^c$ and $d = 0$, the rule can be written as $a^c \to a$. Finally, the rules of type (2), with the form $a^s \to \lambda$, are called forgetting rules.

As usually happens in other P system variants, each neuron has a label. Notation σ_i is used to refer to a neuron labelled by i. Synapses are denoted by (i, j), representing a synapse from σ_i (called the source neuron) to σ_j (called the target neuron).

Graphical representation of an SN P system usually consists in a directed graph describing the initial structure of the system: the nodes correspond to neurons and are labelled after them, while directed arcs correspond to synapses and model the spikes flow. Within neurons, initial number of spikes and rules are drawn. The output neuron is represented with an outgoing synapse which is not connected to any other neuron of the system, meaning that it is connected to the environment. Accordingly, the input neuron is represented with an ingoing synapse which is not connected to any other neuron, meaning that the environment is connected to it. Fig. 1 shows a basic example.

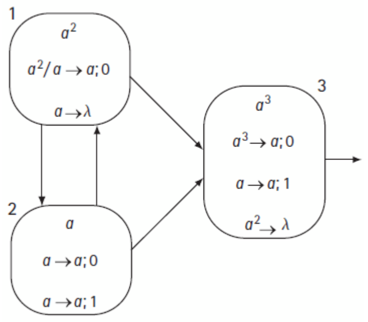

Fig. 1. A simple SN P system.

In what follows, we specify the semantics of SN P systems. SN P systems are synchronized devices. A global clock is assumed, which marks the functioning of the system. SN P systems operate in a non-deterministic maximally parallel way with the following specific feature: at any time instant t, each neuron operates sequentially, since at most only one of the applicable rules over the neuron is applied, which is non-deterministically chosen from the set of applicable rules for the neuron at time instant t. Nevertheless, neurons, as a whole, operate in parallel, since all the neurons with a selected applicable rule fire their rules simultaneously.

Given a neuron σ_i containing a^k spikes, with $k \geq 0$, at a time instant t, it is said that a *firing rule* $E/a^c \to a; d \in R_i$ is *applicable* over σ_i at t if and only if the following conditions hold: (a) σ_i is not executing any rule; (b) $k \geq c$; and (c) $a^k \in L(E)$. Given a neuron σ_i containing a^k spikes, with $k \geq 0$, at a time instant t, it is said that a *forgetting rule* $a^s \to \lambda$ is *applicable* over σ_i at t if and only if the following conditions hold: (a) σ_i is not executing any rule; and (b) $k = s$. Moreover, whenever a neuron is closed executing a firing rule, the rest of (firing/forgetting) rules are disabled.

A neuron checks for applicable rules whenever it receives new spikes or completes a rule execution. Applicable rules are also said to be active or enabled. Neurons having applicable (active, or enabled) rules are said to be active or enabled. It is possible for two firing rules $E_1/a_1^c \to a; d_1$ and $E_2/a_2^c \to a; d_2$ belonging to R_i to become simultaneously applicable, since $L(E_1) \cap L(E_2) \neq \emptyset$. On the other hand, when a forgetting rule $a^s \to \lambda$ belonging to R_i becomes active, only that rule can be applied, since (a) $a^s \notin L(E)$ for any firing rule $E/a^c \to a; d$ belonging to R_i; and (b) the number of spikes in σ_i must be equal to s. When a neuron has more than one applicable rule at instant t, one, and only one, is non-deterministically selected to be applied.

Given a neuron σ_i containing a^k spikes, with $k \geq 1$, at a time instant t, the application of an active firing rule $r \equiv E/a^c \to a; d \in R_i$ over σ_i at t implies the following: (1) at instant t, neuron σ_i fires rule r and c spikes are removed from σ_i immediately, so that $k-c$ spikes are left in the neuron; (2) if $d \geq 1$, from instant t to $t+d-1$, σ_i becomes closed and cannot accept incoming spikes, that is, any spike sent to σ_i in the interval $[t, t+d-1]$ is lost; (3) at instant $t+d$, neuron σ_i becomes open (it accepts incoming spikes) and, simultaneously, it emits a spike (or simply spikes), which is replicated onto the outgoing synapses and sent to the target neurons (spikes reach the target neurons immediately); and (4) at instant $t+d+1$ neuron σ_i can check again for applicable rules.

Given a neuron σ_i containing a^k spikes, with $k \geq 1$, at a time instant t, the application of an active forgetting rule $r \equiv a^s \to \lambda \in R_i$ over σ_i at t implies the following: (1) at instant t, neuron σ_i fires rule r and s spikes are removed from σ_i immediately, so that $k - s$ are left in the neuron, that is, the neuron becomes empty, since $k = s$; and (2) at instant $t+1$ neuron σ_i can check again for applicable rules.

It is worth pointing out that when applying a firing rule with no delay ($d = 0$), the involved neuron is never actually closed: it fires (removing the corresponding spikes) and spikes (sending out a spike) at same instant t, also accepting incoming spikes arriving at that instant t. It is important to distinguish between the terms *firing*, the action of starting a rule application, and *spiking*, the action of sending out a spike. Neurons applying a firing rule both fire and spike (because of this firing rules are also called spiking rules), while neurons applying a forgetting rule fire, but do not spike.

In what follows, we define the concepts of configuration, transition step and computation for SN P systems.

A *configuration* of an SN P system Π at instant i with $i \geq 0$, denoted by C_i, is an instantaneous description of Π at that time instant. A configuration describes, for each neuron in the system, the number of spikes contained in such neuron and the number of time instants left for neuron to become open (zero if already open). The initial configuration of Π is defined as $C_0 = \langle n_1/0, n_2/0, \ldots, n_m/0 \rangle$, meaning that in the initial configuration all the neurons of Π are open.

A *transition step* (or simply step) of an SN P system Π at instant i, with $i \geq 1$, is the state transition of Π from configuration C_{i-1} to C_i, denoted by $C_{i-1} \Rightarrow C_i$. The transition step is performed by selecting and applying rules in a synchronous maximally parallel way as described above. Usually, the term transition step i is used to refer to the transition step taking place at instant i.

A *computation* $\mathcal{C}$ of an SN P system Π is any (finite or infinite) sequence of configurations starting from the initial configuration C_0 such that the i-th term ($i \geq 1$) of the sequence is obtained from the previous one in one transition step. In this way, computation $\mathcal{C}$ is denoted as $\mathcal{C} = C_0 \Rightarrow C_1 \Rightarrow C_2 \Rightarrow \ldots$. If the sequence is infinite, the computation is said to be a non-halting computation. If the sequence is finite, the computation is said to be a halting computation. The last term of a halting computation is a halting configuration, that is, a configuration where all neurons are open and no rules can be applied.

SN P systems can work as number accepting devices, number generating devices, number computing devices and decision problems solvers. For additional information we refer the reader to [21]. Regarding the output of the system, many possibilities exist, such as the number of spikes contained in the output neuron, the number of times that the output neuron spikes, or the difference between the first two time instants when the output neuron spikes.

3 Spiking Neural P Systems Variants

In this section, we discuss some interesting variants that are obtained from classic SN P systems by extending/restricting their syntactical elements or changing the way in which the system operates, that is, their semantics.

3.1 Asynchronous SN P Systems

In this subsection, we discuss some variants of SN P systems working in asynchronous mode. In this mode, neurons with applicable rules at a given computation step may choose not to fire. The "choosing mechanism" may vary, giving place to several asynchronous SN P systems variants.

Asynchronous SN P systems From both mathematical and neurological points of view, it is rather natural to consider asynchronous SN P systems (ASNPS, for short) [7]. In such systems, a global clock, marking the time for all neurons, is still present, but neurons work asynchronously in the following way: the application of rules in each neuron is not obligatory, that is, if a neuron has one or more enabled rules at a given instant, it may (non-deterministically) choose whether or not to fire one of them. If the neuron does not fire, new spikes can come into the neuron rendering some of the previously applicable rules non-applicable. If a rule is still applicable through successive time instants, it can be selected to fire at any time, independently of how much time has passed. Taking this into account, the concepts of configuration, transition step and computation can be defined in a similar way to classic SN P systems.

Limited Asynchronous SN P systems In ASNPS, there is no restriction imposed on the number of successive time instants that a neuron can hold an enabled rule. Nevertheless, from the biological point of view, it is natural to consider a bound imposed on the number of time units that an enabled rule remains unfired, since in nature given a long enough time interval, an enabled chemical reaction will conclude within this interval.

Taking into consideration such biological motivation, limited asynchronous SN P systems (LASNPS, for short) were introduced in [19]. In such systems, a global bound $b \geq 2$ (imposed on all rules) is specified in such a way that if one (and only one) rule in neuron σ_i is enabled at step t and neuron σ_i receives no spike from step t to step $t + b - 2$, then this rule can and must be applied at a step in the next time interval b (that is, at a non-deterministically chosen step from t to $t+b-1$). If the enabled rule in neuron σ_i is not applied, and neuron σ_i receives new spikes, making the rule non-applicable, the computation continues in the new circumstance (maybe other rules are enabled now). If more than one rule is applicable, the neuron non-deterministically chooses and fires one of them in the interval t to $t + b - 1$.

Additionally, there is a special remark with respect to the functioning of LASNPS not explicitly given in [19]: whenever a neuron σ_i has to check for new applicable rules, the count regarding the number of steps that neuron σ_i has been holding the execution of its applicable rules is reset. Let us recall from classic SN P systems that neuron σ_i has to check for new applicable rules whenever it receives new spikes of completing a rule execution. The remark described above implies that, at any instant t, any applicable rule of a given neuron σ_i has been waiting to be applied the same amount of time.

In LASNPS, a configuration is described by the number of spikes present in each neuron, the number of time units for neurons to become open as well as the time that has elapsed for each rule since it became applicable. Transition steps are carried out in a similar way to classic SN P systems, but according to the limited asynchronous firing mechanism described above. Regarding the output of the system, the following applies: since an enabled rule at instant t can be applied at any moment in the time interval t to $t + b - 1$ (that is, in the b steps starting from t), a variable spike train can be produced. Consequently, defining the result of a computation as the number of steps between two consecutive spikes (as usual in synchronous systems) is useless. So, the result of a computation is usually defined as the total number of spikes sent into the environment by the output neuron.

Asynchronous SN P Systems with Local Synchronization In an ASNPS, neurons (asynchronously) fire their rules in an independent way with respect to each other. Nevertheless, from the biological point of view, it is natural to consider the interrelation between neurons in terms of synchronicity. In a biological neural system, neurons involved in carrying out some specific functioning synchronously cooperate with each other to achieve their goals. Groups of neurons like this can exhibit different topologies, such as motifs with 4-

5 neurons and communities with 12-15 neurons. On the other hand, non-related neurons in terms of functionality work in an independent or asynchronous way.

Taking into consideration such biological motivation, asynchronous SN P systems with local synchronization (ASNPSLS, for short) were introduced in [22]. In an ASNPSLS, synchronous interrelations between neurons are described via a family of sets denoted by *Loc*. Each element of the family is a set of locally synchronous neurons, that is, neurons that work synchronously with each other. As such, each element of *Loc* is called a locally synchronous set (ls-set, for short). A neuron can be placed in zero, one or more ls-sets. Given an ASNPSLS with m neurons $\sigma_1, \sigma_2, \ldots, \sigma_m$, the family *Loc* can be formally defined in the following way:

$$Loc = \{loc_1, loc_2, \ldots, loc_l\} \subseteq \mathcal{P}(\{\sigma_1, \sigma_2, \ldots, \sigma_m\}),$$

where $\mathcal{P}(\{\sigma_1, \sigma_2, \ldots, \sigma_m\})$ is the power set of $\{\sigma_1, \sigma_2, \ldots, \sigma_m\}$.

In an ASNPSLS, the behaviour of individual neurons is similar to classic asynchronous SN P systems: at any instant t, a neuron with enabled rules non-deterministically choose whether or not to fire one of such rules. Nevertheless, neurons in the same ls-set fire in a synchronous way: if an enabled neuron within a ls-set loc_j fires, then all neurons in loc_j that have enabled rules must fire at that instant t. As such, it is possible that all neurons from loc_j remain unfired even if they have enabled rules, i.e., all neurons from loc_j may remain still, or all neurons from loc_j with enabled rules fire at a same step. Hence, neurons work asynchronously at the global level, while working synchronously within each ls-set.

The concepts of configuration, transition step and computation can be defined in a similar way to classic ASNPS taking into account the locally synchronous firing mechanism described above.

3.2 SN P Systems with Hybrid Astrocytes

In what follows, we discuss a variant of classic SN P systems taking inspiration from one important biological element existing in the nervous system structure, the astrocyte.

Astrocytes, also known collectively as astroglia, are characteristic star-shaped glial cells in the brain and spinal cord that connect to neighbouring synapses. An astrocyte connects to a synapse in the space between the presynaptic and postsynaptic terminals giving place to the so-called “tripartite synapse” [1], with one single astrocyte being able to connect to different synapses in this way. Astrocytes propagate intercellular Ca_2^+ waves over long distances in response to stimulation and, similarly to neurons, release transmitters (called gliotransmitters) in a Ca_2^+-dependent manner. Moreover, within the dorsal horn of the spinal cord, activated astrocytes have the ability to respond to almost all neurotransmitters [9] and, upon activation, release a multitude of neuroactive molecules that influences neuronal excitability. That is, astrocytes can sense the neuronal activity related with their attached synapses and carry out a synaptic modulation in an excitatory or inhibitory way. Other important functionalities of astrocytes include biochemical support of endothelial cells that form the blood-brain barrier, provision of nutrients to the nervous tissue, maintenance of extracellular ion balance, and a role in the repair and scarring process of the brain and spinal cord following traumatic injuries. Because of all of the above, astrocytes constitute an important area of research within the field of neuroscience and, consequently, they are also an interesting element to take inspiration from regarding the membrane computing paradigm.

In SN P systems with astrocytes, new syntactical ingredients are introduced to model astrocytes. One astrocyte can be attached to one or more synapses and one synapse can be attached to zero, one or more astrocytes. Synapses attached to a given astrocyte are said to be *controlled* by the astrocyte. An astrocyte senses the spike traffic passing along its controlled synapses and can have an excitatory or inhibitory influence on such traffic. In general, excitatory influence implies allowing the spike traffic to go along the controlled synapses, while the inhibitory influence implies destroying such traffic, with the spikes being removed from the system. When a synapse is controlled by two or more astrocytes, spike traffic passing along such synapse only survives when every controlling astrocyte has an excitatory influence on the synapse. In this way, when neurons spike, outgoing spikes are transmitted along synapses and reach target neurons unless they are intercepted by astrocytes.

Several variants of SN P systems with astrocytes have been defined and studied. We here will deal one of them, SN P systems with hybrid astrocytes (SNPSHA, for short).

SNPSHA were introduced in [18]. In these systems, an astrocyte shows an excitatory or inhibitory influence on controlled synapses by comparing the spike traffic on such synapses against a threshold associated with the astrocyte. Since these astrocytes can show either an excitatory or inhibitory influence, they are called *hybrid* astrocytes within the scope of this work (note that they are not called like this in [18]).

Hybrid astrocytes are graphically represented as diamond-shaped figures, with a number inside corresponding to the threshold and lines connected to their controlled synapses (see [18]). An example is shown in Fig. 2.

A hybrid astrocyte can be formally defined in the form $ast_i = (syn_{ast_i}, t_i)$, where syn_{ast_i} is the set of synapses controlled by the astrocyte ast_i, $t_i \in \mathbb{N}$ is the *threshold* of the astrocyte ast_i. Semantics of SNPSHA follows from the classic model, but incorporating hybrid astrocytes behaviour. For an astrocyte ast_j, let us assume that there are k spikes in total passing along the synapses in syn_{ast_j} at an instant t. Then a) if $k > t_j$, the astrocyte ast_j has an inhibitory influence on the neighbouring synapses, and the k spikes are simultaneously suppressed (that is, the spikes are removed from the system); b) if $k < t_j$, the astrocyte ast_j has an excitatory influence on the neighbouring synapses, all spikes survive and get to their destination neurons, reaching them simultaneously; and c) if $k = t_j$, the astrocyte ast_j non-deterministically chooses an inhibitory or excitatory influence on the neighbouring synapses. It is possible that two or more astrocytes control the same synapse. In this case, only if all the controlling astrocytes have an

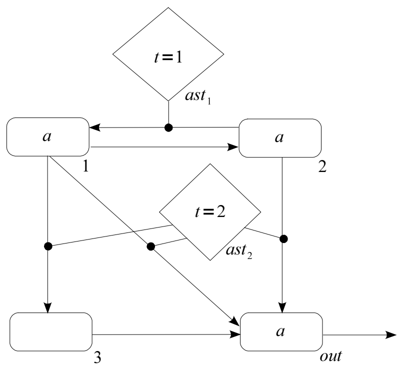

Fig. 2. A spiking neural P system with a hybrid astrocyte.

excitatory influence on the synapse the spikes passing along such synapse can survive and pass to the destination neurons; if one of these astrocytes has an inhibitory influence on the synapse, then the spikes along this synapse are suppressed and removed from the system. Note that in the systems constructed in [18] two astrocytes never control the same synapse.

The concepts of configuration, transition step and computation can be defined in a similar way to classic SN P systems.

3.3 SN P Systems with Anti-spikes

In what follows, we discuss a variant of SN P systems that addresses the inhibitory nature of some neural impulses by introducing an additional object type, the anti-spike, denoted by $\overline{a}$ and somewhat named after anti-matter. Anti-spikes are present in neurons and participate in firing and forgetting rules, along with usual spikes. Spikes and anti-spikes cannot exist simultaneously in the same neuron, since they annihilate each other, as an implicit rule of the form $a\overline{a} \to \lambda$ exists in every neuron. This model was introduced in [17]. Its definition is as follows.

Definition 2. *An SN P system with anti-spikes of degree $m \geq 1$ is a construct of the form:*

$$\Pi = (O, \sigma_1, \sigma_2, \ldots, \sigma_m, syn, in, out),$$

where:

1. *$O = \{a, \overline{a}\}$ is the alphabet (a is called* spike*, $\overline{a}$ is called* anti-spike*);*
2. *$\sigma_1, \sigma_2, \ldots \sigma_m$ are neurons of the form $\sigma_i = (n_i, R_i)$ where:*
   - *$n_i \geq 0$ is the initial number of spikes contained in σ_i;*
   - *R_i is a finite set of rules of the following forms:*
     (0) $a\overline{a} \to \lambda$,
     (1) $E/b^c \to b'; d$ with E a regular expression over $\{a\}$ or over $\{\overline{a}\}$ (but not over a and $\overline{a}$ simultaneously),

(2) $b^s \to \lambda$ with $s \geq 1$ and $b^s \notin L(E)$ for any rule of type (1) in R_i;
verifying that $(b, b') \in \{(a, a), (a, \overline{a}), (\overline{a}, a), (\overline{a}, \overline{a})\}$;

3. *$syn \subseteq \{1, 2, \ldots, m\} \times \{1, 2, \ldots, m\}$, with $(i, i) \notin syn$ for $1 \leq i \leq n$ is the synapse graph, defining the synapses among neurons;*
4. *$in, out \in \{1, 2, \ldots, m\}$ are the input and output neurons respectively;*
5. *for any rule of type (1) $E/b^c \to b'; d \in R_i$, if $i = out$ then $b' = a$.*

Semantics of SN P systems with anti-spikes follows from the classic model, with the following remarks:

– Rules of type (0) are annihilation rules. They are implicitly defined in every neuron of the system, so they are not specified when describing the model. A rule of type (0) is applied with top-most priority in a maximal way, taking zero time units to apply whenever the neuron receives spikes or anti-spikes. Consequently, spikes and anti-spikes cannot exist together in the same neuron.
– Rules of types (1) and (2) are firing and forgetting rules involving spikes and/or anti-spikes, respectively. Applicability and application for these rules are defined as in the classic model, with the peculiarity that regular expressions may involve spikes or anti-spikes (but not both) and that spikes and anti-spikes may be consumed/produced when applying the rules.

The concepts of configuration, transition step and computation can be defined in a similar way to classic SN P systems. With respect to the input and output of the system, only spikes are allowed to be entered into/sent out of the system.

3.4 SN P Systems with Structural Plasticity

In [3,6], an SN P system variant inspired by the biological feature known as *neural plasticity* was introduced, in particular, the so-called *structural plasticity*, with two main mechanisms: (1) synaptogenesis and synapse deletion, and (2) synaptic rewiring [2]. Systems belonging to this variant are called *Spiking Neural P systems with structural plasticity* (SNPSP systems, for short). SNPSP systems variant is a response to the open problem **D** in [20] where "dynamism" only for synapses is to be considered. SNPSP systems are distinct from related variants such as HSNP systems in [8] and the SNP systems in [16,23].

Contrary to classic SN P systems, in SNPSP systems a new class of rules, called *plasticity rules*, are used instead of forgetting rules. Plasticity rules can be defined as follows: $E/a^c \to \alpha k(i, N_j)$, where $c \geq 1$, $\alpha \in \{+, -, \pm, \mp\}$, $k \geq 1$, $1 \leq j \leq |R_i|$, and $N_j \subseteq \{1, \ldots, m\}$ (let us recall that m stands for neuron number). On the other hand, while in SNPSP systems spiking rules are still present, no delays are considered.

In order to describe plasticity rules semantics, the following notation is introduced: given a neuron σ_i, we denote $pres(i) = \{j | (i, j) \in syn\}$ and $pos(i) = \{j | (j, i) \in syn\}$ as the sets of neuron labels having σ_i as presynaptic and postsynaptic neuron, respectively. Plasticity rules are applied as follows. If at instant t neuron σ_i has $b \geq c$ spikes and $a^b \in L(E)$, a rule $E/a^c \to \alpha k(i, N) \in R_i$ can be applied. The set N is a collection of neurons to which σ_i can connect to (synapse creation) or disconnect from (synapse deletion) using the applied plasticity rule. The rule consumes c spikes and performs one of the following processes, depending on α: (a) if either $\alpha = +$ and $N - pres(i) = \emptyset$ or $\alpha = -$ and $pres(i) = \emptyset$, then there is nothing more to do, (c spikes are consumed but no synapse is created or removed); (b) for $\alpha = +$: if $|N - pres(i)| \leq k$, deterministically create a synapse to every σ_l, $l \in N_j - pres(i)$; if however $|N - pres(i)| > k$, then non-deterministically select k neurons in $N - pres(i)$, and create one synapse to each selected neuron; (c) for $\alpha = -$: if $|pres(i)| \leq k$, deterministically delete all synapses in $pres(i)$; if however $|pres(i)| > k$, then non-deterministically select k neurons in $pres(i)$, and delete each synapse to the selected neurons; and (d) if $\alpha \in \{\pm, \mp\}$: create (respectively, delete) synapses at instant t and then delete (respectively, create) synapses at instant $t + 1$. Only the priority of application of synapse creation or deletion is changed, but the application is similar to $\alpha \in \{+, -\}$. The neuron remains open in the interval $[t, t + 1]$, i.e. the neuron can continue receiving spikes. However, the neuron can only apply another rule at instant $t + 2$.

An important remark is that for σ_i applying a rule with $\alpha \in \{+, \pm, \mp\}$, creating a synapse always involves an *embedded* sending of one spike when σ_i connects to a neuron. This single spike is sent at the instant when the synapse creation is applied. Whenever σ_i *attaches* to σ_j using a synapse during synapse creation, we have σ_i immediately transferring one spike to σ_j. Note that the application of rules in neurons are synchronized, that is, a global clock is assumed.

A system state or configuration is denoted $\langle s_1, \ldots, s_m \rangle$ where $s_i, 1 \leq i \leq m$, is the number of spikes contained in σ_i, along with neuron connections based on the synapse graph *syn*, from where $pres(i)$ and $pos(i)$ can be derived, for a given σ_i. The initial configuration is $\langle n_1, \ldots, n_m \rangle$, with the possibility of $syn = \emptyset$. The concepts of transition step and computation can be defined in a similar way to classic SN P systems.

4 Theoretical Results Reference

In what follows, some useful references regarding theoretical results involving the discussed SN P systems variants in this chapter are presented. A good set of useful general references can be found in [10]. Further results on these SN P systems variants can be obtained from: Normal forms [11]; General computational power [13,14,15]; Asynchronous mode [7,19,22]; Hybrid astrocytes [18] and Anti-spike [17]; Structural plasticity [3,4,5,6].

Acknowledgements

L.F. Macías-Ramos, L. Valencia-Cabrera and M.J. Pérez-Jiménez were supported by Project TIN2012-37434 of the Ministerio de Economía y Competitividad of Spain, cofinanced by FEDER funds. L. Pan was supported by National Natural Science Foundation of China (61320106005 and 61472154), Ph.D. Programs Foundation of Ministry of Education of China (2012014213008) and the Innovation Scientists and Technicians Troop Construction Projects of Henan Province (154200510012). H. Adorna was supported by the DOST - Engineering Research and Development for Technology Project, and Semirara Mining Corporation Professorial Chair in Computer Science. F. Cabarle was supported by DOST - Engineering Research and Development for Technology Project.

References

1. Araque, A., Parpura, V., Sanzgiri, R.P., Haydon, P.G.: Tripartite synapses: glia, the unacknowledged partner. Trends in Neuroscience 22(5), 208–215 (1999)
2. Butz, M., Wörgötter, F., van Ooyen, A.: Activity-dependent structural plasticity. Brain Research Reviews 60(2), 287–305 (2009)
3. Cabarle, F., Adorna, H., Ibo, N.: Spiking neural P systems with structural plasticity. In: Pre-proc. of 2nd Asian Conference on Membrane Computing, Chengdu, China. pp. 13 – 26 (2013)
4. Cabarle, F., Adorna, H., Pérez-Jiménez, M.: Asynchronous spiking neural P systems with structural plasticity. In: UCNC2015. (to appear)
5. Cabarle, F., Adorna, H., Pérez-Jiménez, M.: Sequential spiking neural P systems with structural plasticity based on max/min spike number. Neural Computing and Applications (2015), available at: http://dx.doi.org/10.1007/s00521-015-1937-5, (to appear)
6. Cabarle, F., Adorna, H., Pérez-Jiménez, M., Song, T.: Spiking neural P systems with structural plasticity. Neural Computing and Applications (2015), available at: http://dx.doi.org/10.1007/s00521-015-1857-4, (to appear)
7. Cavaliere, M., Egecioglu, O., Ibarra, O.H., Ionescu, M., Păun, G., Woodworth, S.: Asynchronous spiking neural P systems: Decidability and undecidability. In: Garzon, M., Yan, H. (eds.) DNA Computing, Lecture Notes in Computer Science, vol. 4848, pp. 246–255. Springer Berlin Heidelberg (2008)
8. Gutiérrez-Naranjo, M., Pérez-Jiménez, M.: Hebbian learning from spiking neural P systems view. In: et al., D.C. (ed.) WMC9. LNCS, vol. 5391 (2009)
9. Haydon, P.G.: Glia: listening and talking to the synapse. Nature Reviews Neuroscience 2(3), 185–193 (2001)
10. Ibarra, O., Leporati, A., Păun, A., Woodworth, S.: The Oxford Handbook of Membrane Computing, chap. Spiking Neural P Systems. Vol. 1 of Păun et al. [21] (2010)
11. Ibarra, O.H., Păun, A., Păun, G., Rodríguez-Patón, A., Sosík, P., Woodworth, S.: Normal forms for spiking neural P systems. Theoretical Computer Science 372(2-3), 196 – 217 (2007)
12. Ionescu, M., Păun, G., Yokomori, T.: Spiking Neural P systems. Fundam. Inf. 71(2,3), 279–308 (2006)
13. Leporati, A., Mauri, G., Zandron, C., Păun, G., Pérez-Jiménez, M.J.: Uniform solutions to SAT and subset sum by spiking neural P systems. Natural Computing 8(4), 681–702 (2009)
14. Leporati, A., Zandron, C., Ferretti, C., Mauri, G.: Solving numerical NP-complete problems with spiking neural P systems. In: Eleftherakis, G., Kefalas, P., Păun,

G., Rozenberg, G., Salomaa, A. (eds.) Membrane Computing, Lecture Notes in Computer Science, vol. 4860, pp. 336–352. Springer Berlin Heidelberg (2007)

15. Leporati, A., Zandron, C., Ferretti, C., Mauri, G.: On the computational power of spiking neural P systems. IJUC 5(5), 459–473 (2009)
16. Pan, L., Păun, G., Pérez-Jiménez, M.J.: Spiking neural P systems with neuron division and budding. Science China Information Sciences 54(8), 1596–1607 (2011)
17. Pan, L., Păun, G.: Spiking neural P systems with anti-spikes. International Journal of Computers, Communications and Control IV, 273–282 (2009)
18. Pan, L., Wang, J., Hoogeboom, H.J.: Asynchronous extended spiking neural P systems with astrocytes. In: Gheorghe, M., Păun, G., Rozenberg, G., Salomaa, A., Verlan, S. (eds.) Int. Conf. on Membrane Computing. Lecture Notes in Computer Science, vol. 7184, pp. 243–256. Springer Berlin Heidelberg (2011)
19. Pan, L., Wang, J., Hoogeboom, H.J.: Limited asynchronous spiking neural P systems. Fundam. Inform. 110(1-4), 271–293 (2011)
20. Păun, G., Pérez-Jiménez, M.: Spiking neural P systems. recent results, research topics. In: et al., A.C. (ed.) Algorithmic Bioprocesses (2009)
21. Păun, G., Rozenberg, G., Salomaa, A. (eds.): The Oxford Handbook of Membrane Computing. Oxford University Press, Oxford (U.K.) (2010)
22. Song, T., Pan, L., Păun, G.: Asynchronous spiking neural P systems with local synchronization. Information Sciences 219, 197–207 (2013)
23. Wang, J., Hoogeboom, H., Pan, L.: Spiking neural P systems with neuron division. In: M. Gheorghe, e.a. (ed.) CMC 2010. LNCS, vol. 6501. Springer Berlin Heidelberg (2010)
24. Wang, T., Zhang, G., Pérez-Jiménez, M.J.: Fuzzy Membrane Computing: Theory and Applications. International Journal of Computers, Communications and Control, Special Issue on Fuzzy Sets and Applications (Celebration of the 50th Anniversary of Fuzzy Sets) 10(6), 903–934 (2015), http://univagora.ro/jour/index.php/ijccc/article/view/2080