Spiking Neural P Systems: Theoretical Results and Applications

Haina Rong[1], Tingfang Wu[2], Linqiang Pan[2](✉), and Gexiang Zhang[1]

[1] School of Electrical Engineering, Southwest Jiaotong University, Chengdu 610031, China
{ronghaina,zhgxdylan}@126.com

[2] Key Laboratory of Image Information Processing and Intelligent Control of Education Ministry of China, School of Automation, Huazhong University of Science and Technology, Wuhan 430074, Hubei, China
{tfwu,lqpan}@hust.edu.cn

Abstract. Spiking neural P systems, namely SN P systems, are a class of distributed and parallel neural-like computation models, inspired from the way neurons communicate by means of spikes. It has been shown that SN P systems have powerful computation capability and significant potential in real-life applications, and they have received more and more attraction from the scientific community. This paper firstly introduces the formal definition of standard SN P systems and some notions which are often used in this field. Then, the theoretical results about the computation power and efficiency of SN P systems are surveyed. The applications of SN P systems are recalled by summarizing the literature about the real-life applications of SN P systems. Finally, some hot topics and further research lines on SN P systems are provided.

1 Introduction

Natural computing is the computational version of the process of extracting ideas from nature to develop computation systems, or using natural materials (e.g., molecules) to perform computation. Typical examples of natural computing include well-known neural computing inspired by the functioning of the brain, evolution computing inspired by Darwinian evolution of species, swarm intelligence inspired by the collective behavior of groups of organisms, and so on.

Membrane computing is a new branch of natural computing, which was initiated by Gh. Păun in 1998. The area of membrane computing aims to abstract computing ideas from the structure and the functioning of the living cell as well as from the cooperation of cells in tissues, organs, and other populations of cells [38], e.g., constructing computation models and designing algorithms for optimization. The abstracted parallel and distributed computation models in the area of membrane computing are called membrane systems, also called P systems. Since P systems have powerful capability in parallel processing and significant potential in various applications, this area has attracted more and

© Springer Nature Switzerland AG 2018
C. Graciani et al. (Eds.): Pérez-Jiménez Festschrift, LNCS 11270, pp. 256–268, 2018.
https://doi.org/10.1007/978-3-030-00265-7_20

more people's interest. The area of membrane computing was listed by Thompson Institute for Scientific Information (ISI) as an emerging research front in computer science in 2003.

According to the membrane structure, membrane systems fall into three main categories: (1) cell-like P systems which have a hierarchical arrangement of membranes as in a cell; (2) tissue-like P systems which have several one-membrane cells as in a tissue; (3) neural-like P systems which have several neurons as in a neural net.

This paper will survey a kind of neural-like P systems, called spiking neural P systems (SN P systems, for short), which emphasize on a specific type of cell, i.e. the neuron.

The human brain is a complex information processing system, where more than a trillion neurons work in a cooperation manner to perform various tasks. Therefore, the human brain structure and functioning, from neurons, astrocytes, and other components to complex networks, is a gold mine for inspiring efficient computing devices. Inspired by the neurophysiological behavior of neurons sending electrical impulses (spikes) along axons from presynaptic neurons to postsynaptic neurons in a distributed and parallel manner, Ionescu et al. proposed SN P systems in 2006 [19]. SN P systems have become an attractive and promising research direction in the community of membrane computing [61]. Since the introduction of SN P systems, much attention has been paid to various topics, such as the establishment of SN P systems variants from different biological inspirations, computation power, computation efficiency, and applications.

In the past twelve years, there are lots of theoretical results and applications published in journals or conference proceedings. However, a few papers ([35] is one of them in Chinese) focused on the survey of the main results of SN P systems in theory and applications. This paper is to briefly summarize the development and main results achieved in computation models, computing power and efficiency of SN P systems, and the applications in fault diagnosis of electric power systems, combinatorial and engineering optimization, signal classification, etc.

The organization of this paper is as follows. Section 2 reviews the standard SN P system and its various variants. Section 3 summarizes the theoretical results of SN P systems reported in literature. Main applications of SN P systems are presented in Sect. 4. Future research lines are listed in Sect. 5.

2 Spiking Neural P Systems and Variants

A standard spiking neural P system (shortly, SN P system), of degree $m \geq 1$, is a construct of the form

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

where:

- $O = \{a\}$ is the singleton alphabet (a is called spike);

– $\sigma_1, \dots, \sigma_m$ are neurons, of the form

$$\sigma_i = (n_i, R_i), 1 \leq i \leq m,$$

where:

(a) $n_i \geq 0$ is the initial number of spikes contained in the neuron;

(b) R_i is a finite set of rules of the following two forms:

(i) $E/a^c \rightarrow a; d$, where E is a regular expression over $\{a\}$, $c \geq 1$, and $d \geq 0$ is the delay time, that is, the interval between applying the rule and releasing the spike;

(ii) $a^s \rightarrow \lambda$, for some $s \geq 1$, with the restriction that $a^s \notin L(E)$ for any rule $E/a^c \rightarrow a; d$ of type i) from R_i;

– $syn \subseteq \{1, 2, \dots, m\} \times \{1, 2, \dots, m\}$ are synapses between neurons, with $(i, i) \notin syn$ for $1 \leq i \leq m$;

– $in, out \in \{1, 2, \dots, m\}$ indicate the input and output neurons, respectively.

The rules of type are called spiking rules (also called firing rules). A spiking rule $E/a^c \rightarrow a; d$ in neuron σ_i is eanbeld if the following conditions are satisfied: (1) the content of neuron σ_i is described by the regular expression E associated with the rule, e.g., if neuron σ_i contains k spikes, then $a^k \in L(E)$; (2) the number of spikes in neuron σ_i is not less than the number of spikes consumed by the rule, i.e., $k \geq c$. The application of this rule means that neuron σ_i consumes c spikes, and produces one spike after a delay of d steps. If $d = 0$, then the produced one spike emitted by neuron σ_i is replicated and sent immediately to all neurons σ_j such that $(i, j) \in syn$. If $d \geq 1$, assume that the rule is applied at step t, then at steps $t, t+1, \dots, t+d-1$, the neuron is closed, so that it cannot receive new spikes from neurons which have synapses to the closed neuron (if any of these neurons tries to send a spike to the closed neuron, then the particular spike is lost). At step $t + d$, the neuron fires and becomes open again, so that it can receive spikes (which can be used at step $t + d + 1$, when the neuron can apply rules again).

The rules of type (ii) are called forgetting rules. A forgetting rule $a^s \rightarrow \lambda$ in neuron σ_i is eanbeld only if the neuron contains exactly s spikes. The application of such a rule means that neuron σ_i removes all s spikes, but produces no spike.

If a rule $E/a^c \rightarrow a; d$ has $E = a^c$, then it can be written in the simplified form $a^c \rightarrow a; d$.

If a rule $E/a^c \rightarrow a; d$ has $d = 0$, then it can be written in the simplified form $E/a^c \rightarrow a$.

In standard SN P systems, only one spike can be produced by a spiking rule in one time, which is always called a standard spiking rule. With the goal of simplifying the results about universality of SN P systems, Chen et al. proposed extended SN P systems [9], where several spikes can simultaneously be produced by a spiking rule in one time, which is called an extended rule. Specifically, extended rules are of the form $E/a^c \rightarrow a^p; d$, where E is a regular expression over $\{a\}$, $c \geq 1$, and $p, d \geq 0$, with the restriction $c \geq p$. The meaning of such a rule is that if the content of the neuron is described by the regular expression E, then c spikes are consumed and p spikes are produced. Because p can be 0 or greater than 0, extended rules are a generalization of both standard spiking and forgetting rules.

It is assumed that a global clock exists in an SN P system, marking the time for the whole system (for all neurons of the system). In each time unit, if a neuron σ_i have enabled rules, then it must apply (at most) one of the enabled rules. In this way, a situation may appear: neuron σ_i contains two enabled rules $E_1/a^{c_1} \rightarrow a; d_1$ and $E_2/a^{c_2} \rightarrow a; d_2$, and these two rules have $E_1 \cap E_2 \neq \emptyset$. If so, one of them is nondeterministically chosen to be applied. Thus, the rules are used in a sequential manner at the level of each neuron, but neurons function in parallel with each other.

The state of an SN P system at a given time is described by the number of spikes present in each neuron and the number of steps to count down until it becomes open (this number is zero if the neuron is already open). That is, the configuration of system Π is of the form $\langle r_1/t_1, \ldots, r_m/t_m \rangle$ for $r_i \geq 0$ and $t_i \geq 0$, where r_i indicates that neuron σ_i contains r_i spikes, and it will become open after t_i steps, $i = 1, 2, \ldots, m$. With this notation, the initial configuration of system Π is $\langle n_1/0, \ldots, n_m/0 \rangle$. By using the rules as described above, one can get a sequence of consecutive configurations. Each passage from a configuration C_1 to a successor configuration C_2 is called a transition and denoted by $C_1 \Longrightarrow C_2$. Any sequence of transitions starting from the initial configuration constitutes a computation. A computation halts if it reaches a configuration where all neurons are open and no rule is enabled. With any computation, halting or not, one associates a spike train, that is, a binary sequence with occurrences of digit 1 (resp., 0) indicating that the output neuron sends one spike (resp., no spike) out of the system.

When an SN P system works as a number generator, the result of a computation can be defined in several ways. With any spike train containing at least two spikes, the time interval between the first two being emitted is considered as the computation result [19]. This way of defining the result of a computation has been extended in [37]: generalizing to the first the time intervals between the first k spikes of a spike train, or the time intervals between all consecutive spikes, or only alternately the time intervals between all consecutive spikes, etc.

The way of defining the result of a computation in membrane computing can also be introduced in SN P systems. That is, one can also consider the result of a computation as the total number of spikes sent into the environment by the output neuron when the computation halts [6].

An SN P system can also work as a number acceptor [19]. In general, a number is introduced in the form of the time interval between two spikes entering the system. If the computation eventually halts, then this number is said to be accepted by the system.

Moreover, the result of a computation can be also defined as the spike train itself [7]. In this way, an SN P system is used as a binary string language generator defined on the binary alphabet $\{0, 1\}$.

3 Theoretical Results of Spiking Neural P Systems

These results mainly concern two aspects: computation power and computation efficiency. We start by briefly presenting some results of the first type.

For standard SN P systems, Ionescu et al. proved that these systems are Turing universal (also say computationally complete), that is, equivalent with Turing machines, when they are used as number generators or number acceptors [19]. Moreover, if a bound is given on the number of spikes present in any neuron along a computation, then a characterization of semilinear sets of numbers is obtained; and a characterization of finite sets can be obtained by standard SN P systems with one neuron.

From both mathematical and computer science points of view, it is always desirable to make the construction of SN P systems as simple as possible without the loss of computing power, i.e. a normal form. Ibarra et al. first investigated the influence of some ingredients of SN P systems on the computation power of these systems, such as the regular expressions used in spiking rules, the delay in spiking rules, and forgetting rules [18]. It was proved that in the case of removing the delay in spiking rules or forgetting rules, standard SN P systems are Turing universal. Afterwards, Pan et al. improved these results and proved that standard SN P systems are still Turing universal with the restrictions: (1) the delay in spiking rules and forgetting rules are not used, (2) each neuron contains at most two rules, and (3) the rules in the neurons using two rules have the same regular expression which controls their firing [28].

The resource (here, in terms of the number of neurons) needed for constructing universal computing devices of various types has been heavily investigated in computer science, e.g., Siegelmann et al. constructed a universal recurrent neural network by using 886 neurons [44]. This issue was also considered in SN P systems. Gh. Păun et al. constructed a universal SN P system having 49 neurons in the case of using extended rules and having 84 neurons in the case of using standard rules for SN P systems used as a device of computing functions; as a number generator, a universal SN P system with standard rules having 76 neurons, and one with extended rules having 50 neurons were obtained [36]. The comparison result shows that SN P systems have a "desired" computation power while using less resource. Following the research line of small universal computing devices, some bio-inspired features and mathematical strategies have been introduced into universal SN P systems for computing natural numbers and functions, in order to reduce the number of neurons. Some known results are shown in Table 1.

Besides used as number generators/acceptors and function computing devices, SN P systems can also be used as language generators. In a standard SN P system, the output neuron sends at most one spike into the environment in one time, thus the time instances when one spike exits the output neuron are marked with the digit 1, and the time instances when no spike is emitted by the output neuron are marked with the digit 0. In this way, the system can generate the binary string languages defined on the binary alphabet $\{0, 1\}$. In this definition, it was proved that the generative capacity of standard SN P systems

Table 1. The known results of Turing universal SN P systems with small numbers of neurons.

	Number of neurons for computing functions	Number of neurons for computing natural numbers	Type of rules
Bio-inspired features			
Weighted synapses [34]	38	36	Standard
Rules on synapses [47]	30	-	Extended
	39	-	Standard
Rules on synapses with high capacity neurons [50]	-	6	Extended
Request rules [46]	28	4	Extended
Colored spikes [49]	-	3	Extended
Mathematical strategies			
Combined modules [65]	41	41	Extended
	67	63	Standard
Cooperating rules	59 [25]	-	Standard
	-	8 [45]	Extended
High capacity neurons	-	10 [32]	Extended
"Super" neurons with an infinite number of rules [26]	-	4	Extended

is rather restricted, even some finite languages cannot be generated by these systems. However, by using morphism and projection, standard SN P systems can generate recursively enumerable languages [7].

For extended SN P systems, because several spikes can exit at the same time, the time instances when the output neuron emits i spikes are marked with the symbol b_i. In this way, extended SN P systems can generate any language defined on an arbitrary alphabet. It was proved that extended SN P systems can directly generate recursively enumerable languages [9].

In standard SN P systems, all neurons function in parallel at the level of the system (i.e., the system works in the synchronous manner), while at most one rule can be applied at the level of each neuron (i.e., the system works sequentially) [19]. Inspired by computer science theories, SN P systems working in different modes were investigated, such as non-synchronized (i.e., asynchronous) mode [6], that is, a neuron can apply or not apply its rules in any step; sequential mode: at each step of a computation, one (resp. all) of the neurons with the maximum/minimum number of spikes among the neurons that are active will fire [17,22]; exhaustive mode: whenever a rule is enabled in a neuron, it is used as many times as possible for the number of spikes from that neuron [20,33].

With the inspirations of different biological phenomena, various new types of SN P systems were proposed. For example, inspired by the functioning of inhibitory impulses among biological neurons, Pan et al. introduced anti-spikes and inhibitory synapses into SN P systems, and proved a series of normal forms of SN P systems with anti-spikes [27,48]. Inspired by the fact that astrocytes play an important role in the functioning and interaction of neurons, and astrocytes have excitatory and inhibitory influence on synapses, Hoogeboom et al. proposed SN P systems with astrocytes [31]. It was proved that SN P systems with astrocytes using simple neurons (all neurons have the same rule, one per neuron, of a very simple form) can achieve Turing universality. With the goal of identifying an easy way to determine the applicability of rules, Wang et al. proposed SN P systems with weights, which are inspired by the fact that a biological neuron can fire only when its membrane potential reaches or exceeds its threshold potential [51]. It was proved that SN P systems with weights are universal. Motivated by the excitatory or inhibitory nature of Ranvier nodes in biological neurons, Chen et al. constructed axon P systems [10]. Afterwards, Zhang et al. proved that four nodes (respectively, nine nodes) are enough for axon P systems to achieve Turing universality as number generators (respectively, function computing devices) [64]. Based on such a neurobiological fact that in the chemical synapse transmitting, there are multiple ion channels in a synapse, Peng et al. proposed SN P systems with multiple channels, and proved that such systems are universal [41].

Moreover, with mathematical and computer science motivations, many new types of SN P systems were also proposed. Based on the self-organizing and self-adaptive feature of artificial neural networks, Cabarle et al. introduced structural plasticity into the framework of SN P systems [4]. SN P systems with structural plasticity were proved to be universal. Afterwards, Song et al. proved that SN P systems with structural plasticity without any synapse at the beginning of a computation can also achieve Turing universality [57]. Incorporating ideas from nonstatic (i.e. dynamic) graphs and networks, Cabarle et al. proposed SN P systems with scheduled synapses, where synapses in such systems are available only at a specific schedule or duration [3]. SN P systems with scheduled synapses were proved to be universal. With mathematical motivation, Wu et al. consider a combination of basic features of cell-like P systems and of SN P systems, that is, consider cell-like P systems with only one kind of objects and spiking rules as those in SN P systems, called cell-like SN P systems [59]. It was proved that cell-like SN P systems with four membranes can achieve Turing universality. In order to simplify the integration-and-fire conditions of SN P systems, Wu et al. exploited polarizations $+, 0, -$ to control the application of rules instead of regular expressions [58]. It was proved that SN P systems with three kinds of polarizations are universal. Inspired by the way that components communicate with each other by a request-response pattern in parallel-cooperating grammar systems, Pan et al. proposed SN P systems with communication on request [30]. It was proved that SN P systems with communication on request are universal when two types of spikes are used.

Another topic on SN P systems is to study their computation efficiency, that is, to study whether SN P systems can solve computationally hard problems in a feasible time. Under the assumption that $P \neq NP$, Leporati et al. proved that a deterministic SN P system of a polynomial size cannot solve an NP-complete problem in a polynomial time, i.e., Milano theorem for SN P systems [14]. Hence, some features need to be introduced into SN P systems in order to enhance the efficiency of such systems. Generally, there are three kinds of features introduced into SN P systems to solve computationally hard problems as follows.

(1) Pre-computed resources (here, in terms of exponential number of neurons): Chen et al. first exploited SN P systems with pre-computed resources to solve in a constant time the NP-complete problem SAT in a semi-uniform way [8]. Leporati et al. provided semi-uniform and uniform solutions to the numerical NP-complete problem Subset Sum by using SN P systems with exponential size pre-computed resources [23]. Afterwards, Ishdorj et al. shown that the two PSPACE-complete problems QSAT and Q3SAT can be solved in a polynomial time by SN P systems with pre-computed resources in a uniform way. All the systems constructed above work in a deterministic way [21].
(2) Nondeterminism: Leporati et al. provided the solutions to SAT problem and Subset Sum problem in a semi-uniform or uniform way by using nondeterministic SN P systems but without pre-computed resources [14]. In [13], standard SN P systems without the delay feature and having a uniform construction were obtained.
(3) Neuron division and neuron budding: inspired by neural stem cell division, Pan et al. introduced the features of neuron division and neuron budding into the framework of SN P systems, which can generate exponential workspace in linear time. It was shown that SN P systems with neuron division and neuron budding solve computationally hard problems by means of a space-time tradeoff in a polynomial time, which is illustrated with an efficient solution to SAT problem [29]. Afterwards, Wang et al. exploited SN P systems only with neuron division to provide a uniform solution to SAT problem in a polynomial time [52].

4 Applications of Spiking Neural P Systems

This section presents an overview of SN P systems from the perspective of real-life applications with respect to engineering optimization, fault diagnosis of electric power systems, image processing, and signal identification.

The design of optimization algorithms can build a bridge between SN P systems and real-life applications. In [62], an extended SN P system (ESNPS) was proposed by introducing the probabilistic selection of evolution rules and multi-neurons output and a family of ESNPS, called optimization SN P system (OSNPS), were further designed through using a guider to adaptively adjust rule probabilities to approximately solve combinatorial optimization problems. In [54], OSNPS was used to solve the fault section estimation problem in an electric power system by formulating it into an optimization problem. Thus, OSNPS can search output fault sections in an automatic way when the status information of protective relays and circuit breakers coming from a supervisory control and data acquisition system is considered as the input. Several types of fault cases consisting of a single fault, multiple faults and multiple faults with incomplete and uncertain information in an electric power system can be accurately diagnosed in the simulation experiments.

The application for diagnosing the faults in an electric power system is an attractive research direction and is well investigated in the past years through introducing an algebraic fuzzy reasoning approach into SN P systems called fuzzy reasoning SN P systems (FRSNPS) [39,55]. The most attractive aspect of FRSNPS is that they can offer an intuitive illustration based on a strictly mathematical expression, a good fault-tolerant capacity due to its handling of incomplete and uncertain messages in a parallel manner, a good description for the relationships between protective devices and faults, and an understandable diagnosis model-building process [56]. Until now, FRSNPS have been successfully used to diagnose the faults existing in transformers [39], power transmission networks [40,56], traction power supply systems of high-speed railways [53], metro traction power systems [16], fault classification of power transmission lines [43] and fault lines detection in a small current grounding system [42].

As many problems in the processing of digital images have features which make it suitable for techniques inspired by nature, in [12], a novel link between SN P systems and Digital Imagery was presented, by providing an implementation of the parallel Guo and Hall algorithm in SN P systems to solve the skeletonization problem.

The learning ability for SN P systems is the fundamentals to be used for classification. In [15], the first attempt was made by using the framework of SN P systems to implement Hebbian learning. In [11], SN P systems were used to identify nuclear export signals and the promising experimental result with accurate rate 74.18 % was obtained.

Software for experimental simulations have been developed for SN P systems and their variants, as in [1,2,5,24,60].

5 Concluding Remarks and Future Research Lines

By introducing some further biologically-inspired features into SN P systems, it is worth developing new computation models which are "more realistic" to get closer to the brain. Moreover, to bring enough further biologically-inspired features to SN P systems may possibly model and simulate processes taking place in the "real" brain.

Most of the aforementioned SN P systems were proved to be Turing universal. In the area of SN P systems, a challenging and interesting problem is to look for classes of SN P systems which are not equivalent with Turing machines, but also not computing only semilinear sets of numbers. From the point of view of theory, such classes of systems are rather significant, because of the possibility of finding classes of systems with decidable properties. Moreover, from the point of view of applications, these classes of systems are also attractive, because of the possibility of finding properties of the modeled processes by analytical, algorithmic means.

The computation efficiency of SN P systems deserves further efforts. It is interesting to introduce new ingredients in the area of SN P systems to generate an exponential workspace in polynomial time, by trading space for time in solving **NP**-complete problem and **PSPACE**-complete problem.

There are many ingredients of usual P systems which were not considered for SN P systems, e.g., promoters, inhibitors, membrane creation. Thus, with mathematical motivation, considering these ingredients in the framework of SN P systems might also make sense.

Future application work on SNP systems could be focused on the *killer real applications* [63]. There are many possibilities like fault diagnosis with FRSNPS, OSNPS for engineering optimization and classification with the SNP systems with learning ability. Several important topics on FRSNPS were listed in [56]. The extension of OSNPS to numerical optimization problems is an ongoing task. How to develop a learning network with SNP systems is also a challenging topic.

Acknowledgments. This work was supported by National Key R&D Program of China for International S&T Cooperation Projects (No. 2017YFE0103900), National Natural Science Foundation of China (61320106005, 61672437, 61702428, and 61772214) and Sichuan Science and Technology Program (2018GZ0085, 2018GZ0185, 2017GZ0159).

References

1. Barbuti, R., Maggiolo-Schettini, A., Milazzo, P., Tini, S.: Compositional semantics of spiking neural P systems. J. Log. Algebr. Program. **79**(6), 304–316 (2010)
2. Cabarle, F.G.C., Adorna, H.N.: On structures and behaviors of spiking neural P systems and petri nets. In: Csuhaj-Varjú, E., Gheorghe, M., Rozenberg, G., Salomaa, A., Vaszil, G. (eds.) CMC 2012. LNCS, vol. 7762, pp. 145–160. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36751-9_11
3. Cabarle, F.G.C., Adorna, H.N., Jiang, M., Zeng, X.: Spiking neural P systems with scheduled synapses. IEEE Trans. Nanobioscience (2017). https://doi.org/10.1109/TNB.2017.2762580
4. Cabarle, F.G.C., Adorna, H.N., Pérez-Jiménez, M.J., Song, T.: Spiking neural P systems with structural plasticity. Neural Comput. Appl. **26**(8), 1905–1917 (2015)
5. Carandang, J.P., Villaflores, J.M.B., Cabarle, F.G.C., Adorna, H.N., Martinez-del-Amor, M.A.: CuSNP: spiking neural P systems simulators in CUDA. Rom. J. Inf. Sci. Technol. **20**(1), 57–70 (2017)
6. Cavaliere, M., Ibarra, O.H., Păun, G., Egecioglu, O., Ionescu, M., Woodworth, S.: Asynchronous spiking neural P systems. Theor. Comput. Sci. **410**(24–25), 2352–2364 (2009)
7. Chen, H., Freund, R., Ionescu, M., Păun, G., Pérez-Jiménez, M.J.: On string languages generated by spiking neural P systems. Fundam. Inform. **75**(1–4), 141–162 (2007)
8. Chen, H., Ionescu, M., Ishdorj, T.O.: On the efficiency of spiking nNeural P systems. In: Proceedings of the 8th International Conference on Electronics, Information, and Communication, Ulan Bator, Mongolia, pp. 49–52 (2006)

9. Chen, H., Ionescu, M., Ishdorj, T.O., Păun, A., Păun, G., Pérez-Jiménez, M.J.: Spiking neural P systems with extended rules: universality and languages. Nat. Comput. **7**(2), 147–166 (2008)
10. Chen, H., Ishdorj, T.O., Păun, G.: Computing along the axon. Prog. Nat. Sci. **17**(4), 417–423 (2007)
11. Chen, Z., Zhang, P., Wang, X., Shi, X., Wu, T., Zheng, P.: A computational approach for nuclear export signals identification using spiking neural P systems. Neural Comput. Appl. **29**(3), 695–705 (2018)
12. Díaz-Pernil, D., Peña-Cantillana, F., Gutiérrez-Naranjo, M.A.: A parallel algorithm for skeletonizing images by using spiking neural P systems. Neurocomputing **115**, 81–91 (2013)
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. Nat. Comput. **8**(4), 681–702 (2009)
14. Leporati, A., Zandron, C., Ferretti, C., Mauri, G.: On the computational power of spiking neural P systems. Int. J. Unconv. Comput. **5**, 459–473 (2009)
15. Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J.: A spiking neural P system based model for Hebbian learning. In: Frisco, P., Corne, D.W., Păun, Gh. (eds.) Proceedings of WMC9, Edinburgh, UK, pp. 189–207 (2008)
16. He, Y., Wang, T., Huang, K., Zhang, G., Pérez-Jiménez, P.J.: Fault diagnosis of metro traction power systems using modified fuzzy reasoning spiking neural P systems. Rom. J. Inf. Sci. Technol. **18**(3), 256–272 (2015)
17. Ibarra, O.H., Păun, A., Rodríguez-Patón, A.: Sequential SNP systems based on min/max spike number. Theor. Comput. Sci. **410**(30–32), 2982–2991 (2009)
18. Ibarra, O.H., Păun, G., Păun, G., Rodríguez-Patón, A., Sosík, P., Woodworth, S.: Normal forms for spiking neural P systems. Theor. Comput. Sci. **372**(2–3), 196–217 (2007)
19. Ionescu, M., Păun, G., Yokomori, T.: Spiking neural P systems. Fundam. Inform. **71**(2), 279–308 (2006)
20. Ionescu, M., Păun, G., Yokomori, T.: Spiking neural P seystems with an exhaustive use of rules. Int. J. Unconv. Comput. **3**(2), 135–153 (2007)
21. Ishdorj, T.O., Leporati, A., Pan, L., Zeng, X., Zhang, X.: Deterministic solutions to QSAT and Q3SAT by spiking neural P systems with pre-computed resources. Theor. Comput. Sci. **411**(25), 2345–2358 (2010)
22. Jiang, K., Song, T., Pan, L.: Universality of sequential spiking neural P systems based on minimum spike number. Theor. Comput. Sci. **499**, 88–97 (2013)
23. Leporati, A., Gutiérrez-Naranjo, M.A.: Solving subset sum by spiking neural P systems with pre-computed resources. Fundam. Inform. **87**(1), 61–77 (2008)
24. Macías–Ramos, L.F., Pérez–Hurtado, I., García–Quismondo, M., Valencia–Cabrera, L., Pérez–Jiménez, M.J., Riscos–Núñez, A.: A P–Lingua Based Simulator for Spiking Neural P Systems. In: Gheorghe, M., Păun, G., Rozenberg, G., Salomaa, A., Verlan, S. (eds.) CMC 2011. LNCS, vol. 7184, pp. 257–281. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28024-5_18
25. Metta, V.P., Raghuraman, S., Krithivasan, K.: Small universal spiking neural P systems with cooperating rules as function computing devices. In: Gheorghe, M., Rozenberg, G., Salomaa, A., Sosík, P., Zandron, C. (eds.) CMC 2014. LNCS, vol. 8961, pp. 300–313. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-14370-5_19
26. Neary, T.: A boundary between universality and non-universality in extended spiking neural P systems. In: Dediu, A.-H., Fernau, H., Martín-Vide, C. (eds.) LATA

2010. LNCS, vol. 6031, pp. 475–487. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13089-2_40

27. Pan, L., Păun, G.: Spiking neural P systems with anti-spikes. Int. J. Comput. Commun. Control **4**(3), 273–282 (2009)
28. Pan, L., Păun, G.: Spiking neural P systems: an improved normal form. Theor. Comput. Sci. **411**(6), 906–918 (2010)
29. Pan, L., Păun, G., Pérez-Jiménez, M.J.: Spiking neural P systems with neuron division and budding. Sci. China Inf. Sci. **54**(8), 1596–1607 (2011)
30. Pan, L., Păun, G., Zhang, G., Neri, F.: Spiking neural P systems with communication on request. Int. J. Neural Syst. **27**(08), 1750042 (2017)
31. Pan, L., Wang, J., Hoogeboom, H.J.: Spiking neural P systems with astrocytes. Neural Comput. **24**(3), 805–825 (2012)
32. Pan, L., Zeng, X.: A note on small universal spiking neural P systems. In: Păun, G., Pérez-Jiménez, M.J., Riscos-Núñez, A., Rozenberg, G., Salomaa, A. (eds.) WMC 2009. LNCS, vol. 5957, pp. 436–447. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-11467-0_29
33. Pan, L., Zeng, X.: Small universal spiking neural P systems working in exhaustive mode. IEEE Trans. Nanobioscience **10**(2), 99–105 (2011)
34. Pan, L., Zeng, X., Zhang, X., Jiang, Y.: Spiking neural P systems with weighted synapses. Neural Process. Lett. **35**(1), 13–27 (2012)
35. Pan, L., Zhang, X., Zeng, X., Wang, J.: Research advances and prospect of spiking neural P systems. Chin. J. Comput. **31**(12), 2090–2096 (2008). (in Chinese)
36. Păun, A., Păun, G.: Small universal spiking neural P systems. BioSystems **90**(1), 48–60 (2007)
37. Păun, G., Pérez-Jiménez, P.J., Rozenberg, G.: Spike trains in spiking neural P systems. Int. J. Found. Comput. Sci. **17**(04), 975–1002 (2006)
38. Păun, G., Rozenberg, G., Salomaa, A. (eds.): The Oxford Handbook of Membrane Computing. Oxford University Press, Inc., New York (2010)
39. Peng, H., Wang, J., Pérez-Jiménez, M.J., Wang, H., Shao, J., Wang, T.: Fuzzy reasoning spiking neural P system for fault diagnosis. Inf. Sci. **235**, 106–116 (2013)
40. Peng, H., et al.: Fault diagnosis of power systems using intuitionistic fuzzy spiking neural P systems. IEEE Trans. Smart Grid (2017). https://doi.org/10.1109/TSG.2017.2670602
41. Peng, H., et al.: Spiking neural P systems with multiple channels. Neural Netw. **95**, 66–71 (2017)
42. Rong, H., Ge, M., Zhang, G., Zhu, M.: A novel approach for detecting fault lines in a small current grounding system using fuzzy reasoning spiking neural P systems. Int. J. Comput. Commun. Control **13**(2), 458–473 (2018)
43. Rong, H., Zhu, M., Feng, Z., Zhang, G., Huang, K.: A novel approach for fault classification of power transmission lines using singular value decomposition and fuzzy reasoning spiking neural P systems. Rom. J. Inf. Sci. Technol. **20**(1), 18–31 (2017)
44. Siegelmann, H.T., Sontag, E.D.: On the computational power of neural nets. J. Comput. Syst. Sci. **50**(1), 132–150 (1995)
45. Song, T., Pan, L.: A small universal spiking neural P system with cooperating rules. Rom. J. Inf. Sci. Technol. **17**(2), 177–189 (2014)
46. Song, T., Pan, L.: Spiking neural P systems with request rules. Neurocomputing **193**, 193–200 (2016)
47. Song, T., Pan, L., Păun, G.: Spiking neural Psystems with rules on synapses. Theor. Comput. Sci. **529**, 82–95 (2014)

48. Song, T., Pan, L., Wang, J., Venkat, I., Subramanian, K.G., Abdullah, R.: Normal forms of spiking neural P systems with anti-spikes. IEEE Trans. Nanobioscience **11**(4), 352–359 (2012)
49. Song, T., Rodríguez-Patón, A., Zheng, P., Zeng, X.: Spiking neural P systems with colored spikes. IEEE Trans. Cogn. Dev. Syst. (2017). https://doi.org/10.1109/TCDS.2017.2785332
50. Song, T., Xu, J., Pan, L.: On the universality and non-universality of spiking neural P systems with rules on synapses. IEEE Trans. NanoBioscience **14**(8), 960–966 (2015)
51. Wang, J., Hoogeboom, H.J., Pan, L., Păun, G., Pérez-Jiménez, M.J.: Spiking neural P systems with weights. Neural Comput. **22**(10), 2615–2646 (2010)
52. Wang, J., Hoogeboom, H.J., Pan, L.: Spiking neural P systems with neuron division. In: Gheorghe, M., Hinze, T., Păun, G., Rozenberg, G., Salomaa, A. (eds.) CMC 2010. LNCS, vol. 6501, pp. 361–376. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-18123-8_28
53. Wang, T., Zhang, G., Pérez-Jiménez, M.J., Cheng, J.: Weighted fuzzy reasoning spiking neural P systems: application to fault diagnosis in traction power supply systems of high-speed railways. J. Comput. Theor. Nanosci. **12**(7), 1103–1114 (2015)
54. Wang, T., Zeng, S., Zhang, G., Pérez-Jiménez, M.J., Wang, J.: Fault section estimation of power systems with optimization spiking neural P systems. Rom. J. Inf. Sci. Technol. **18**(3), 240–255 (2015)
55. Wang, T., Zhang, G., Pérez-Jiménez, M.J.: Fuzzy membrane computing: theory and applications. Int. J. Comput. Commun. Control **10**(6), 904–935 (2015)
56. Wang, T., Zhang, G., Zhao, J., He, Z., Wang, J., Pérez-Jiménez, P.J.: Fault diagnosis of electric power systems based on fuzzy reasoning spiking neural P systems. IEEE Trans. Power Syst. **30**(3), 1182–1194 (2015)
57. Wang, X., Song, T., Gong, F., Zheng, P.: On the computational power of spiking neural P systems with self-organization. Sci. Rep. **6**, 27624 (2016)
58. Wu, T., Păun, A., Zhang, Z., Pan, L.: Spiking neural P systems with polarizations. IEEE Trans. Neural Netw. Learn. Syst. (2017). https://doi.org/10.1109/TNNLS.2017.2726119
59. Wu, T., Zhang, Z., Păun, G., Pan, L.: Cell-like spiking neural P systems. Theor. Comput. Sci. **623**, 180–189 (2016)
60. Zeng, X., Adorna, H., Martínez-del-Amor, M.Á., Pan, L., Pérez-Jiménez, M.J.: Matrix representation of spiking neural P systems. In: Gheorghe, M., Hinze, T., Păun, G., Rozenberg, G., Salomaa, A. (eds.) CMC 2010. LNCS, vol. 6501, pp. 377–391. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-18123-8_29
61. Zhang, G., Pérez-Jiménez, M.J., Gheorghe, M.: Real-life Applications with Membrane Computing. Springer, Berlin (2017)
62. Zhang, G., Rong, H., Neri, F., Pérez-Jiménez, M.J.: An optimization spiking neural P system for approximately solving combinatorial optimization problems. Int. J. Neural Syst. **24**(5), Article no. 1440006 (2014)
63. Zhang, G., et al.: Real applications of membrane computing models. In: Gheorghe, M., et al. (eds.) Multidisciplinary Creativity: Homage to Gheorghe Paun on his 65th Birthday, pp. 173–185 (2015)
64. Zhang, X., Pan, L., Păun, A.: On the universality of axon P systems. IEEE Trans. Neural Netw. Learn. Syst. **26**(11), 2816–2829 (2015)
65. Zhang, X., Zeng, X., Pan, L.: Smaller universal spiking neural P systems. Fundam. Inform. **87**(1), 117–136 (2008)