Bachelor's and master's degree thesis proposals

Some of these topics are based on Spiking Neural P Systems (SNPS), a model introduced in [14], which provides a general overview of its functioning; others concern Spiking Neural Networks (SNN), and a recent review can be found in [1]. For a clearer and more concise description that includes SNPS, SNN, and some possible applications, see [37].

A simulator capable of executing SNPS with anti-spike and white hole (see [30]) is currently under development; at this **link**, some of the proposed topics involve this code. For a practical approach to working on an SNN that analyzes images, a comprehensive tutorial can be found **here**. Different topics may share the same papers, as they are connected by common themes.

1 Federated learning

The work in [33] provides an in-depth explanation of how Federated Learning (FL) operates when applied to SNNs, with practical examples, code, and numerous experimental results. The topic is of great interest because it combines two emerging areas: distributed learning and neural models inspired by biological functioning.

In [32], although not focused on SNNs, several research directions in the field of green FL are discussed, summarized in Figure 1. Techniques such as sparsification make it possible to obtain graphs with a significantly smaller number of edges than the maximum possible, making them applicable to both SNNs and SNPS. However, in the case of SNNs, synapses have real-valued weights to transmit, while in SNPS they are boolean connections (presence or absence of synapses), making the compression problem conceptually different.

Other techniques, such as pruning and quantization, can be explored to reduce model complexity, and these topics are related to the themes of graph compression. A possible thesis direction could therefore involve comparing the main efficiency techniques of Federated Learning applied to ANN, SNN, and SNPS, assessing their benefits in terms of computational and spatial savings, as well as their adaptability to spiking models.

A simple implementation of an SNPS can be found at this **link**, later evolved into **this one**, specifically structured to be privacy-preserving (ideal for FL) and connected to the work in [25]. However, it is incomplete and not formal, showing that there is little research in this direction.

Another research direction concerns the modification of distributed learning

algorithms between client and server for SNNs. Such networks require specific methods, such as Backpropagation Through Time (BPTT) or Batch Normalization Through Time (BNTT), which differ significantly from the algorithms used in ANNs. These methods also need to be adapted to realistic scenarios where:

- data are not independent and identically distributed (non-IID), and classes are unbalanced;
- the energy and computational resources of devices are limited;
- training time or costs must be reduced in distributed environments.

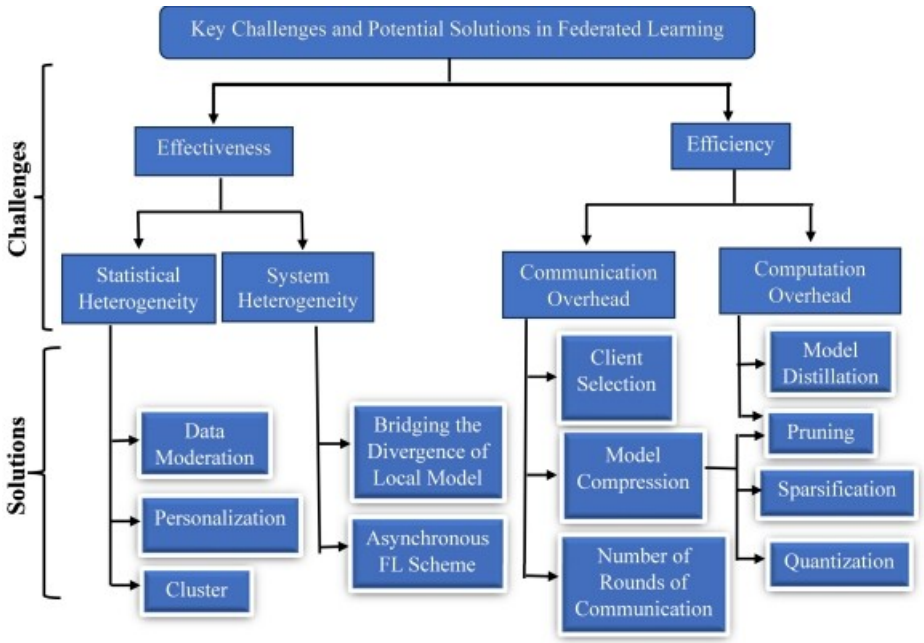


Figure 1: Challenges and possible solutions in Federated Learning toward a more sustainable approach [32].

Research example

- Find, in the paper [32] or other similar ones, an area where FL and SNNs offer advantages. For example, in wearable devices, where information must remain private and energy consumption has major constraints;
- Look for more specific papers on this topic, such as [29], and study the database used;

- Implement an equivalent or similar model, study it, and pursue the research directions proposed by the chosen paper.

2 Tuning of SNPS with heuristic algorithms

Currently, in the **simulator** mentioned earlier, a simple network composed of three layers has been implemented, but its structure is fully customizable: it is possible to change the number of layers and included neurons, the arrangement of synapses, and the internal rules of the neurons.

The goal is to achieve higher performance through parameter tuning of the system.

It is also possible to intervene on synaptic connectivity, removing or inhibiting specific connections to train and improve the model. Unlike classical artificial neural networks, where optimization acts on synaptic weights, SNPS require different mechanisms, and new strategies to manage this dynamic are currently under exploration.

In this context, the use of heuristic optimization algorithms comes into play. Therefore, a tuning phase is required, consisting of three main subphases.

- Structure tuning: The number of layers and neurons per level can be modified, as well as the function performed by each layer (max/average pooling, convolution, edge detection, fully connected). The problem is complex because neurons have adaptable activation rules, and several different architectures can be built. This process can be addressed, for example, through Tabu Search or by defining a neighborhood function that specifies small changes to the network topology, such as adding or removing neurons or synapses.

- Rule tuning: Rule tuning can be managed using Simulated Annealing or similar techniques to identify the optimal combination of parameters. In this case, too, the search space is extremely large: each neuron may have multiple rules, and each rule presents several values to optimize. The problem is complex due to the high number of variables involved.

- Synapse tuning: To describe this phase, imagine a triangular search area where each vertex corresponds to the percentage of inhibited synapses (negative behavior), deleted synapses, and maintained synapses, summing to 1. Each combination produces different levels of accuracy, and the exploration of this space can be performed using Particle Swarm Optimization or similar methods.

The model is new and still little studied, which brings some challenges but also many opportunities for improvement, learning, and innovative research through the use of unconventional optimization techniques.

See [24] for a general overview of learning techniques in this model, and [8] for a survey on image processing techniques in membrane computing (including tissue-like and Spiking Neural approaches).

The work in [41], later extended in [43], proposes a model in which multiple Extended SNPS operate in parallel, with modifiable rule application probabilities, leading to an Optimization SNPS capable of solving the Knapsack Problem. This represents a significant example of how SNPS can be adapted and optimized to tackle a specific theoretical problem, with a concrete and well-defined tuning phase.

Research example

- Analyse the paper just cited [41] and its extension [43];
- Recreate the code presented by abstracting from the membrane model, i.e. simulating it with matrices or equivalent techniques;
- Find a related research direction. For example, test on a knapsack problem with a more complex profit function; or on a different NP problem, or differentiate between the exploration and exploitation phases;
- Adapt the code, obtain results and compare them with the original work in terms of performance, structure, realism and model complexity.

3 Minimization of energy costs

After an initial phase of accurately estimating the computational costs associated with system execution, the next step would be to define the objective function and study mechanisms to reduce it. The network structure can be modeled as a directed graph, where nodes represent neurons and edges represent synapses. Each edge has an associated fixed weight, corresponding to the energy cost required to transmit a spike, while each node has a weight representing the energy needed for firing rule applications (potentially higher in the presence of multiple rules). These operations may include regular expression checks, counting, or internal state manipulations; therefore, it is necessary to distinguish between computational cost and actual energy cost (in joules), allowing later translation from the former to the latter.

The objective is to minimize the total cost of the network while maintaining functional correctness, evaluated through metrics such as accuracy or other comparable performance measures. The optimization process can be addressed using local search or graph-based metaheuristic techniques, comparing the different resulting structures in terms of costs, benefits, and limitations.

A challenge in this research direction lies in the absence of hardware devices that allow the direct implementation and use of SNPS, making precise and reliable estimation of energy costs difficult. A possible approach is to adapt these models to neuromorphic chips developed for SNNs, such as Loihi 2, or to a lesser extent SpiNNaker or TrueNorth. Further information on works published using the Loihi 2 chip can be found **here** .

When applied to SNNs, this thesis topic is more explored, since the reduction of energy cost in SNNs compared to ANNs is well documented. The work presented in [31] uses quantization-aware training in an SNN for image classification and compares its energy consumption with that of traditional training, showing how SNN energy usage can be analyzed.

Research example

- Deepen knowledge of the energy costs of SNNs by studying the main scientific works that describe their characteristics;
- Analyse which layers, activation functions and mechanisms of those networks have higher or lower energy consumption;
- Identify a publication that proposes a network of medium-low complexity and reproduce its implementation as described by the authors;
- Reduce the energy costs of the network and compare the new results with the previous ones, aiming to minimise performance degradation and maximise energy benefits;
- Apply the methods adopted to similar networks, analysing their response to similar changes.

4 Equivalences between models

A possible research direction involves developing software capable of translating an SNPS model into other computational formalisms. For example, in [19] Spiking Petri Nets are introduced, along with an algorithm to convert them into SNPS and another to perform the reverse transformation. In [3], a correspondence between the two models is also outlined — these are just a few examples. Following an initial theoretical analysis, it is possible to identify equivalence relations (or weaker forms of correspondence) between SNPS and discrete event models, finite or timed automata, logic circuits, and others.

Subsequently, one could design a translator that serves as a bridge between the chosen models. This would have a dual value: on one hand, it would contribute to a deeper understanding of the formal properties of SNPS (even though these have already been extensively studied); on the other hand, it would enable connections with models that are more widespread and practically applicable. In this way, SNPS could become an intermediate modeling language from which to derive representations that are more easily implementable or simulatable in hardware or software environments.

An interesting reference in this direction is [5], where, although the focus is on tissue-like P systems rather than SNPS, the VHDL language was used to implement a P system on FPGA, demonstrating the possibility of transposing the model into physical hardware.

Research example

- Identify a theoretical model similar to SNPS, possibly already related to them, as in the case of Virus Machines described in [28];
- Study the identified model and, where possible, establish a direct parallelism between its components and the SNPS model;
- Analyse variants of SNPS that allow any gaps to be filled, in order to apply the theory developed so far to these variants and also to the model under consideration;
- Implement code that allows conversion from one model to another in the most efficient way possible, minimising the number of neurons, rules and synapses in the SNPS.

5 Evolution of Simulators

The WebSnapse simulator [11], available at this **link** , allows users to create and execute small SNPS through an intuitive web interface, making it particularly useful for educational and outreach activities. A possible work direction, more suited to an internship, consists of developing an extension of the simulator using the code provided in **this** GitHub repository.

An improvement to the simulator could involve introducing mechanisms for the evolution of rules, aimed at exploring optimization dynamics of P systems. It is therefore proposed to implement a framework inspired by genetic algorithms, allowing the automatic evolution of P system rules and configurations, with fitness metrics related to system correctness or efficiency. See the code available in **this** repository, which presents a genetic algorithm applied to SNPS.

Research example

- Try out the WebSnapse simulator and analyse its code;
- Learn about the use of Genetic Algorithms in the field of P systems. For example, in this paper [6], a GA is used to reduce the size of SNPS, while in this work [9], it is used to generate possible SNPS that satisfy a specific task;
- Choose a similar research approach and test various types of genetic algorithms to improve SNPS in various aspects such as performance, number of neurons and number of rules present.

6 Biological inspiration

The human brain represents an extremely efficient processing system, capable of performing a huge number of operations in parallel with very low energy

consumption compared to traditional computational systems. Although it is not optimal in terms of speed or numerical precision, it excels in tasks such as visual recognition, language processing, and associative learning, thanks to the highly distributed and adaptive nature of its neural networks. An example is the speed with which the human visual system extracts spatial and semantic information from a complex scene, identifying objects, relationships, and context with only a few milliseconds of latency.

A possible research direction is to model, within the framework of SNPS or SNN, specific biological mechanisms such as those of the retina, the visual cortex, or other sensory areas. The goal is not the full reproduction of a functioning brain, but the simulation of well-understood local processes — for example, the behavior of retinal ganglion cells or the neuronal competition mechanisms that lead to edge recognition [18].

Such studies require at least a basic understanding of the underlying neurophysiological principles and a willingness to explore neuroscience literature. A comprehensive work that covers SNNs and brain-inspired computing is [39].

Research example

- Investigate in detail the functioning of a specific human visual mechanism;
- Simulate its behaviour with a spiking model;
- Analyse its effectiveness and the results obtained by comparing them with the actual output of the retina.

7 Parallel, discrete and continuous computation

To increase model performance and move closer to the actual functioning of P systems, the use of the GPU is essential. Each membrane in the P system computes independently and in parallel, providing advantages in both efficiency and fidelity with respect to the theoretical behavior. The first step would therefore be adapting the simulator for a parallel approach, using frameworks such as CUDA, OpenCL, or PyTorch to manage simultaneous membrane computation. This first phase could serve as a possible internship project focused on implementation and optimization.

Another direction of study concerns time-dependent computation in SNNs. In theoretical Spiking Neural Network models, time is continuous, and spikes occur at real-valued instants; however, in practical simulations on digital hardware, time is discretized into small intervals (Δt), updating membrane potentials and synapses at each step. This discretization allows the network to be executed synchronously and in parallel, approximating continuous behavior and enabling efficient GPU use.

In SNPS models, on the other hand, time is completely discrete and synchronous: rules are applied simultaneously to all neurons, defining a well-structured sequence of computational steps. Comparing these approaches makes

it possible to analyze two different conceptions of computational time — one continuous and biologically inspired, the other discrete and formal — and to assess advantages, limitations, and possible integrations. More information on this comparison can be found in [24].

A possible thesis could therefore focus on analyzing parallel implementations of temporal computation in spiking models (SNN and SNPS). The main objectives would include:

- implementing a parallel SNPS architecture executable on GPU, with independent computation per membrane;
- studying differences in time representation between SNNs (continuous time) and SNPS (discrete and synchronous time), analyzing their impact on performance and correctness;
- exploring hybrid models, such as Layered SNPS [40, 21], where weights or fuzzy activations are introduced, comparing the benefits these fusions bring compared to the original models;
- proposing a comparison framework that enables transitions between different spiking computation paradigms, highlighting emerging properties and potential advantages in terms of parallelism and temporal modeling.

See this **GitHub** repository as a starting point for code related to SNPS.

Research example

A possible thesis could therefore focus on analysing the parallel implementation of temporal computation in both spiking models. The main objectives would include:

- Implementing a parallel SNPS architecture, executable on GPUs, with independent computation per membrane. See the work [38] for a matrix-based definition of SNPS;
- Study the differences in time representation between SNN (continuous time) and SNPS (discrete and synchronous time), analysing their impact on performance and correctness;
- Explore hybrid models, see for example Layered SNPS [40, 21], in which fuzzy weights or activations are introduced, comparing the advantages brought by these fusions compared to the original models;
- Propose a comparison framework that allows switching between different spiking computation paradigms, highlighting their emerging properties and possible advantages in terms of parallelism and temporal modelling.

8 Different models of SNPS

In the developed simulator, only the extension based on anti-spikes has been implemented, but many other variants exist that could be integrated, making the system more flexible and adaptable. Different types of P systems, together with their respective simulators, are described in [2] and, in more detail, in section 2.2 of [24].

The goal of this line of work is to enable the simulation of different model variants within the same framework, allowing direct comparisons between them. It will be necessary to identify which variants are the most promising and worthy of further exploration through an initial phase of literature research and analysis of their potential applications.

Without this preliminary study phase, the activity might be more suitable for an internship, as it would be more focused on the implementation aspect of the project.

Research example

- Investigate a specific variant of SNPS, for example P systems with Polarization [35];
- Analyse how this variant can be integrated into the **simulator** under development;
- Work in your own GitHub branch, subsequently unifying the proposed variant and observing its functionality.

9 Self organizing maps

It is proposed to extend the SNPS model by introducing a spatial component, so that each neuron is associated with coordinates (x, y), or possibly (x, y, z), representing its position in space. The synapses would then connect neurons that are physically close to each other, allowing the simulation of biological or anatomical structures (for example, a lung, a heart, or parts of the brain) in a spatial manner, integrating concepts from graph drawing.

To make the model dynamic, rules could be introduced that allow neurons to modify their position or the length of their synapses in response to stimuli. A longer synapse could represent a weaker connection, therefore closer to breaking. In this way, the system would be able to self-organize in space based on its internal activity: this idea forms the basis of the concept of **Self Organizing Maps** (SOM).

The work [20] presents a short review of SOMs and their main uses, with reference also to [10], where SOMs are applied to the recognition of medical images using a dedicated software tool, which is now partially outdated but still relevant as a starting point.

In [12], a model is presented that uses SNNs and integrates an SOM to classify simple images (the digits of the MNIST dataset, 28×28 pixels), providing a concrete example of how a spiking approach can promote unsupervised self-organization.

There are therefore two possible research directions: on one hand, to further develop Spiking SOMs (related to SNNs); on the other, to explore the possibility of an SNPS with positional values and rules that allow movement, thus bringing the model closer to the self-organizing behavior of SOMs.

Research example

- Deepen knowledge of the model with papers such as [34], which add the possibility of creating and destroying synapses thanks to rules;
- Program this feature so that it can be used to dynamically change the topological structure of the network;
- Find a research area and a corresponding dataset where this feature can be useful; areas can range from galaxy assembly [13] to the selection of species representative of ecological communities [22];
- Analyse the behaviour of the network applied to the chosen problem, comparing it with similar works.

10 Liquid state machine

The Liquid State Machine (LSM), introduced in [17] and part of the **Reservoir Computing** framework, is described intuitively on the **linked website**, which provides code examples and practical approaches to its implementation. They are entirely based on SNNs and are designed to effectively capture information that changes over time, as shown in [23]. This makes them less suitable for static datasets, such as radiographic images, but very useful for analyzing sequences of images showing temporal evolution — for instance, tumor growth or repeated scans of the same anatomical region.

Several open-source codes are available for their simulation, for example **this one** or **this one**, which also includes a concise explanation of their operation.

A possible development would be to deepen the understanding of this type of model, with the goal of analyzing in detail the behavior of the SNN that constitutes it. It will be necessary to identify datasets suitable for the task (temporally variable) and design a structure for their classification.

An interesting aspect is highlighted in [42], which proposes a combination of LSM and SOM.

11 Digital twin

Digital Twins (DT) are not simple simulators, but digital models that maintain a continuous and bidirectional link with the real system they represent. They receive real-time data from sensors or images, dynamically updating their state, and in turn, they can send decisions or predictions to the physical system. In this sense, the DT "lives" alongside its real counterpart, rather than merely reproducing it statically.

SNNs naturally fit this paradigm, as they process temporal data asynchronously and are inherently energy efficient. As discussed in [15], the integration between neuromorphic computing and Digital Twin Technology (DTT) can improve real-time processing, energy efficiency, and model adaptability, opening new perspectives for dynamic and scalable systems.

Possible applications include the creation of biomedical digital twins, for example, for monitoring cardiac or brain activity [36], where the spiking network learns and replicates physiological patterns in real time, predicting their evolution and detecting potential anomalies.

This still little-explored research direction proposes using SNNs as the dynamic "brain" of a Digital Twin, combining temporal modeling with the predictive and adaptive capabilities typical of neuromorphic computing.

12 Privacy-preserving SNPS

A possible research direction concerns the design and training of SNPS networks that are robust and resistant to privacy attacks. This topic is partly related to Federated Learning (FL), though not entirely overlapping. The works in [16, 25, 26] address privacy issues in spiking networks, while [27] provides a more detailed and clear discussion of vulnerabilities and possible solutions.

The code available on **GitHub** , already mentioned in the section dedicated to FL, simulates a privacy attack. However, it does not represent a formal SNPS: the implemented model includes several abstractions absent from the classical definition of the system, such as softmax and argmax functions, explicit synaptic weights, and linear applications of the form $x \cdot w + b$, typical of artificial neural networks.

It can therefore be stated that while the theoretical foundations of these approaches have been explored, practical implementations consistent with the SNPS model are still lacking. This represents a promising research direction, aimed at bridging the gap between theoretical formalism and real-world applications in the context of privacy-preserving computation.

13 Hybrid models

Another possible research direction involves combining different models or designing hybrid versions that represent a sensible compromise between SNNs, SNPS, and others. Such solutions open the door to numerous possibilities, both

as extensions of existing models and as new variants. In any case, the approach is always to study its behaviour from a theoretical and mathematical point of view, implement its operation with a simulator, and analyse its performance. Some proposals are outlined below.

- **SNPS with weights**: The work [26] presented interesting results in the field of cybersecurity, using a modified SNPS that includes weights in specific layers. In [40], a similar model is used, called the Layered SN P System (LSNPS), which is weighted and fuzzy, for image classification. It is worth noting that the code in **this link**, cited in the section on FL, was provided by the authors of that paper. Similar models could be analyzed in depth, studying their structure and proposing improvements in terms of system efficiency or robustness.

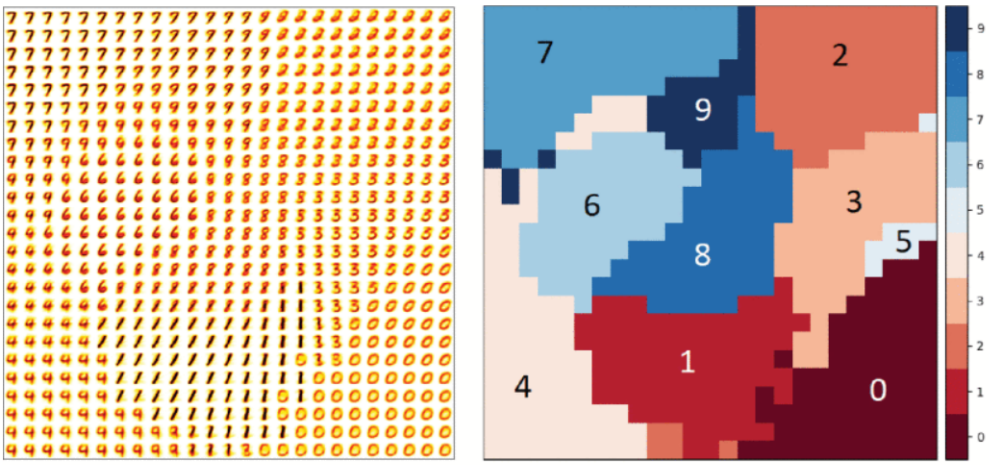
- **SNPS with structural plasticity**: SNPS with structural plasticity are similar to the code developed in the **simulator**, where some synapses are destroyed or inhibited during training. These models include specific rules to govern such behavior, and in their homogeneous version, they use identical rules for all neurons. Although most works focus on formal language theory rather than practical applications, they are still relevant because they rigorously formalize concepts that have been informally explored in the simulator. See [4, 7] for further details.

- **ANN + SNPS**: It is possible to design hybrid models combining artificial neural networks (ANN) and SNPS, for example using CNNs for feature extraction and SNPS for classification, with the goal of achieving greater energy efficiency in this latter stage.

- **SNN/SNPS + GAN**: These architectures could be useful in the medical field — for example, to simulate the temporal evolution of a tumor or to generate realistic scenarios for prognostic analysis. It is proposed to use Generative Adversarial Networks (GAN) to create temporal sequences showing the progression of nodules and pathologies. An approach integrating SNNs or SNPS with GANs could enable a more biologically plausible generation of temporal data.

- **SOM + GAN**: Using a SOM-SNN such as the one described in [12], it would be possible to associate tissue classes (e.g., healthy, diseased, or degenerating) instead of digits from 0 to 9. As shown in Figure 2, in the case of MNIST, digits are arranged in an orderly way in the feature space. Similarly, a SOM-SNN could be structured to arrange neurons in a way that reconstructs, for instance, the shape of a lung. In this scenario, clustering would not only serve to group similar classes but also to recreate a coherent spatial structure. Since neurons are connected by synapses, positional information would remain traceable, allowing the evaluation of how the initial configuration evolved to fit the learned image. It is a complex and experimental idea but could open interesting research directions

— especially if combined with a GAN capable of generating realistic lung images, for example.



(a) 25x25 lattice of neuron filters (b) Neuron class assignments

Figure 2: The graph in the figure is taken from the SOM in [12], with similar results also reported in [42].

Bibliography

[1] Sales G Aribe Jr. "Spiking Neural Networks: The Future of Brain-Inspired Computing". In: ().

[2] Francis George C Cabarle. "Thinking about spiking neural P systems: some theories, tools, and research topics". In: *Journal of Membrane Computing* 6.2 (2024), pp. 148–167.

[3] Francis George C Cabarle and Henry N Adorna. "On structures and behaviors of spiking neural P systems and Petri nets". In: *International Conference on Membrane Computing*. Springer. 2012, pp. 145–160.

[4] Francis George C Cabarle et al. "Spiking neural P systems with structural plasticity". In: *Neural Computing and Applications* 26.8 (2015), pp. 1905–1917.

[5] Javier Carnero, Daniel Dıaz-Pernil, and Miguel A Gutiérrez-Naranjo. "Designing tissue-like P systems for image segmentation on parallel architectures". In: *Ninth Brainstorming Week on Membrane Computing* 2011 (2011), pp. 43–62.

[6] Lovely Joy P Casauay et al. "A Framework for Evolving Spiking Neural P Systems." In: *International Journal of Unconventional Computing* 16 (2021).

[7] Ren Tristan A de la Cruz et al. "Homogeneous spiking neural P systems with structural plasticity". In: *Journal of Membrane Computing* 3.1 (2021), pp. 10–21.

[8] Daniel Díaz-Pernil, Miguel A Gutiérrez-Naranjo, and Hong Peng. "Membrane computing and image processing: a short survey". In: *Journal of Membrane Computing* 1.1 (2019), pp. 58–73.

[9] Jianping Dong et al. "Automatic design of spiking neural P systems based on genetic algorithms". In: *International Journal of Unconventional Computing* 16.2-3 (2021), pp. 201–216.

[10] L Grajciarova et al. "Biomedical image analysis using self-organizing maps". In: *Matlab Conference*. 2012.

[11] Mutya Gulapa et al. "Websnapse reloaded: The next-generation spiking neural p system visual simulator using client-server architecture". In: *Workshop on Computation: Theory and Practice (WCTP 2023)*. Atlantis Press. 2024, pp. 434–461.

[12] Hananel Hazan et al. "Unsupervised learning with self-organizing spiking neural networks". In: *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2018, pp. 1–6.

[13] Benne W Holwerda et al. "Galaxy and mass assembly (GAMA): Self-Organizing Map application on nearby galaxies". In: *Monthly Notices of the Royal Astronomical Society* 513.2 (2022), pp. 1972–1984.

[14] Mihai Ionescu, Gheorghe Păun, and Takashi Yokomori. "Spiking neural P systems". In: *Fundamenta informaticae* 71.2-3 (2006), pp. 279–308.

[15] Vijayakumar Kempuraj and C Lakshmi. "A Review of Neuromorphic Computing and Its Potential for Enhancing Digital Twin Technology". In: *International Conference on Computational Intelligence in Data Science*. Springer. 2025, pp. 224–242.

[16] Youngeun Kim, Yeshwanth Venkatesha, and Priyadarshini Panda. "Privatesnn: privacy-preserving spiking neural networks". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 36. 1. 2022, pp. 1192–1200.

[17] Wolfgang Maass, Thomas Natschläger, and Henry Markram. "Real-time computing without stable states: A new framework for neural computation based on perturbations". In: *Neural computation* 14.11 (2002), pp. 2531–2560.

[18] Richard H Masland. "The neuronal organization of the retina". In: *Neuron* 76.2 (2012), pp. 266–280.

[19] Venkata Padmavati Metta, Kamala Krithivasan, and Deepak Garg. "Spiking Neural P systems and Petri nets". In: *Proc. of the Int'l Workshop on Machine Intelligence Research*. 2009.

[20] Dubravko Miljković. "Brief review of self-organizing maps". In: *2017 40th international convention on information and communication technology, electronics and microelectronics (MIPRO)*. IEEE. 2017, pp. 1061–1066.

[21] Hai Nan et al. "Competitive Nonlinear Layered Spiking Neural P System for solving classification problems". In: *Neurocomputing* 637 (2025), p. 130036.

[22] Young-Seuk Park et al. "Application of a self-organizing map to select representative species in multivariate analysis: a case study determining diatom distribution patterns across France". In: *Ecological Informatics* 1.3 (2006), pp. 247–257.

[23] Meet K Patel et al. "Image Classification using Convolution Liquid State Machines". In: *2025 3rd International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)*. IEEE. 2025, pp. 2084–2091.

[24] Prithwineel Paul, Petr Sosik, and Lucie Ciencialova. "A survey on learning models of spiking neural membrane systems and spiking neural networks". In: *arXiv preprint arXiv:2403.18609* (2024).

[25] Mihail-Iulian Plesa, Marian Gheorghe, and Florentin Ipate. "Privacy-preserving Linear Computations in Spiking Neural P Systems". In: *arXiv preprint arXiv:2309.13803* (2023).

[26] Mihail-Iulian Pleșa et al. "Applications of spiking neural P systems in cybersecurity". In: *Journal of Membrane Computing* 6.4 (2024), pp. 310–317.

[27] Mihail-Iulian Pleşsa, Marian Gheoghe, and Florentin Ipate. "Private Inference on Layered Spiking Neural P Systems". In: *International Work-Conference on the Interplay Between Natural and Artificial Computation*. Springer. 2024, pp. 163–172.

[28] Antonio Ramírez-de-Arellano, David Orellana-Martín, and Mario J Pérez-Jiménez. "Bridges between spiking neural membrane systems and virus machines". In: *International Journal of Neural Systems* 34.06 (2024), p. 2450034.

[29] Yonglin Ren and Gang Zhao. "A Novel Spiking Neural Network for Wearable-Based Human Activity Recognition". In: *2024 6th International Conference on Frontier Technologies of Information and Computer (ICFTIC)*. IEEE. 2024, pp. 1657–1661.

[30] Tao Song et al. "Spiking neural P systems with white hole neurons". In: *IEEE transactions on nanobioscience* 15.7 (2016), pp. 666–673.

[31] Martino Sorbaro et al. "Optimizing the energy consumption of spiking neural networks for neuromorphic applications". In: *Frontiers in neuroscience* 14 (2020), p. 662.

[32] Dipanwita Thakur et al. "Green federated learning: A new era of green aware AI". In: *ACM Computing Surveys* 57.8 (2025), pp. 1–36.

[33] Yeshwanth Venkatesha et al. “Federated learning with spiking neural networks”. In: *IEEE Transactions on Signal Processing* 69 (2021), pp. 6183–6194.

[34] Xun Wang et al. “On the computational power of spiking neural P systems with self-organization”. In: *Scientific reports* 6.1 (2016), p. 27624.

[35] Tingfang Wu et al. “Spiking neural P systems with polarizations”. In: *IEEE transactions on neural networks and learning systems* 29.8 (2017), pp. 3349–3360.

[36] Hui Xiong et al. “The digital twin brain: A bridge between biological and artificial intelligence”. In: *Intelligent Computing* 2 (2023), p. 0055.

[37] Claudio Zandron. “An Overview on Applications of Spiking Neural Networks and Spiking Neural P Systems”. In: *Languages of Cooperation and Communication: Essays Dedicated to Erzsébet Csuhaj-Varjú to Celebrate Her Scientific Career* (2025), pp. 267–278.

[38] Xiangxiang Zeng et al. “Matrix representation of spiking neural P systems”. In: *International conference on membrane computing*. Springer. 2010, pp. 377–391.

[39] Yi Zeng et al. “Braincog: A spiking neural network based, brain-inspired cognitive intelligence engine for brain-inspired ai and brain simulation”. In: *Patterns* 4.8 (2023).

[40] Gexiang Zhang et al. “A layered spiking neural system for classification problems”. In: *International journal of neural systems* 32.08 (2022), p. 2250023.

[41] Gexiang Zhang et al. “An optimization spiking neural P system for approximately solving combinatorial optimization problems”. In: *International Journal of Neural Systems* 24.05 (2014), p. 1440006.

[42] Youdong Zhang et al. “Unsupervised spiking neural network based on liquid state machine and self-organizing map”. In: *Neurocomputing* 620 (2025), p. 129120.

[43] Ming Zhu et al. “An adaptive optimization spiking neural P system for binary problems”. In: *International Journal of Neural Systems* 31.01 (2021), p. 2050054.