Properties of SN P system and its Configuration Graph

Henry N. Adorna

Department of Computer Science (Algorithm and Complexity)
University of the Philippines Diliman
1101 Quezon City, Philippines
E-mail: hnadorna@up.edu.ph

Abstract. Several studies have been reported in the literature about SN P system and its variants. Often, the results provide universality of various variants and the classes of languages that these variants generate and recognize. The state of SN P system is its configuration. We refer to our previous result on reachability of configuration as the Fundamental state equation for SN P system. This paper provides a preliminary investigation on the behavioral and structural properties of SN P system without delay that depend primarily to this fundamental state equation. Also, we introduce the idea of configuration graph $CG_H$ of an SN P system $H$ without delay to characterize behavioral properties of $H$ with respect to $CG_H$. The matrix $M_H$ of an SN P system $H$ without delay is used to characterize structural properties of $H$.

Keywords: Membrane Computing, Spiking Neural P System, Matrices, Configuration Graphs, Behavioral and Structural Properties

1 Introduction

In the 2006, Spiking Neural P system [19] was introduced as a neural-like P system that is an additional to cell-like and tissue-like types of P system models in Membrane computing [26]. Several results have been reported about SN P system including its different variants, particularly proving universality and characterization of languages they recognize and generate [5, 17]. Others considered application areas [6, 12, 31] and simulations of SN P systems in silico [4, 10, 20, 22]. Survey of results on SN P systems (until 2016) is reported in [24].

In the 2010, Zeng et al. [30] introduced a matrix representation of a Spiking Neural P (SN P) system (without delay). Several in silico implementations of SN P systems [4, 10, 20, 22] are based on the matrix representation of [30] including the matrix representation of some variants of P systems [3, 21].

Recently, [1], defined reachability of configuration of SN P systems using the solvability of the matrix equation defined in [30]. This equation plays significant and fundamental role in this paper, so that we refer to it as follows,
**Fundamental state equation for SN P systems**

Let $\Pi$ be an SN P system. Let $k \in \mathbb{Z} \cup \{0\}$. Then

\[
C^{(k+1)} = C^{(k)} + S_p^{(k)} \cdot M_{\Pi}
\]

where $C^{(k+1)}$ and $C^{(k)}$ are the $(k+1)^{th}$ and $k^{th}$ configurations, respectively. $S_p^{(k)}$ is a valid spiking vector obtained from $C^{(k)}$, and $M_{\Pi}$ is the matrix representation of $\Pi$.

Also, in [1], $\text{Struct-}M_{\Pi}$ is defined as a square matrix associated with $\Pi$ that considers only (directed) connectivity of neurons by synapses. $\Pi$ has a cycle, if $\text{row-rank}(\text{Struct-}M_{\Pi}) < m$, where $m$ is the number of neurons in $\Pi$. This idea of considering only the graph structure of SN P systems was mentioned earlier in [13] in considering periodicity in SN P system.

In this paper, we look closely into the graph representation of computation by SN P systems. We investigate on the behavioral and structural properties of SN P systems using this graph representation. We introduce and define the concept of a configuration graph $CG_{\Pi}$ for SN P system $\Pi$ without delay based on equation (1). This concept is similar to the derivation tree that describes the computation of a formal grammar and that of the computation trees associated with P system described in [7]. The configuration graph $CG_{\Pi}$ of SN P system $\Pi$ without delay is a tuple $CG_{\Pi} = (V, E)$ where $V(CG_{\Pi})$ is the set of all reachable configurations of $\Pi$ and $E(CG_{\Pi})$ is the set of pairs $(C^{(i)}, C^{(j)})$ such that $C^{(j)} = C^{(i)} + S_p^{(i)} \cdot M_{\Pi}$, for some valid spiking vector $S_p^{(i)}$. The graph $CG_{\Pi}$ is an edge-labeled directed graph whose labels are the valid spiking vectors. We do initial investigation on some petri net-like properties of SN P system $\Pi$ as defined in [3, 8, 15, 23, 29] with respect to its configuration graph $CG_{\Pi}$. These properties of SN P system $\Pi$ that depends on its (fixed) initial configuration will be distinguished as behavioral properties as opposed to those that only rely on the topology or connectivity of neurons by synapses. The properties of SN P systems that do not depend on (a fixed) initial configurations are called structural properties. Structural properties of SN P system $\Pi$ are investigated here using its matrix representation $M_{\Pi}$.

This paper is a preliminary work on the studies of behavioral and structural properties of SN P systems without delay. This continues the explorations of the matrix representation of SN P systems started in [1, 20]. We provided required definitions and concepts needed for this work in Section 2. The idea of configuration graph is presented in Section 3. Petri net-like properties of SN P systems without delay are defined in Section 4. Then we proceed to Section 5 to present our results characterizing behavioral and structural properties of SN P system $\Pi$ without delay via configuration graphs and matrix representation of $\Pi$, respectively. We finish the paper with some remarks and observations in Section 6.
2 Preliminaries

We assume the readers to be familiar with membrane computing and its neural-like type of P system as presented in [27], and [28], respectively. The basic understanding of linear algebra, particularly, on vectors and matrix operations will be useful.

Definition 1. An SN P system without delay of degree \( m \), \( m \geq 1 \), is a tuple

\[
\Pi = (O, \sigma_1, \ldots, \sigma_m, \text{syn}, \text{in}, \text{out}),
\]

where

- \( O = \{a\} \) is a singleton alphabet (\( a \) is called spike);
- \( \sigma_i, 1 \leq i \leq m \), are neurons, \( \sigma_i = (n_i, R_i), 1 \leq i \leq m \), where
  - \( n_i \geq 0 \) is the number of spikes in \( \sigma_i \);
  - \( R_i \) is a finite set of rules of the following forms:
    * (Type (1); spiking rules)
      \( E/a^c \rightarrow a^p \); where \( E \) is a regular expression over \( \{a\} \), and \( c \geq 1 \), \( p \geq 1 \), such that \( c \geq p \);
    * (Type (2); spiking rules)
      \( a^s \rightarrow \lambda \), for \( s \geq 1 \), such that for each rule \( E/a^c \rightarrow a^p \) of type (1) from \( R_i \), \( a^s \notin L(E) \);
- \( \text{syn} = \{(i, j) \mid 1 \leq i, j \leq m, i \neq j\} \) (synapses between neurons);
- \( \text{in}, \text{out} \in \{1, \ldots, m\} \) indicate the input and output neurons respectively.

The SN P system \( \Pi \) computes by applying one rule at a time from each neuron, but the executions happen simultaneously at the same time. In this work, all SN P systems are only those without delay, unless stated otherwise.

Definition 2. (Configuration Vector of SN P systems \( \Pi \) without delay)

A configuration of SN P system \( \Pi \) is an \( m \)-size vector of integers

\[
C = (a_1, a_2, \ldots, a_m),
\]

where \( a_j \), \( 1 \leq j \leq m \), represents the number of spikes in neuron \( \sigma_j \).

A configuration at time \( k \), \( k \geq 0 \), of an SN P system \( \Pi \), as above, is a vector

\[
C^{(k)} = (a_1^{(k)}, a_2^{(k)}, \ldots, a_m^{(k)}),
\]

where \( a_j^{(k)} \in \mathbb{Z}^+ \cup \{0\}, 1 \leq j \leq m \), is the number of spikes present at time \( k \) in neuron \( \sigma_j \).

The vector \( C^{(0)} = (a_1^{(0)}, a_2^{(0)}, \ldots, a_m^{(0)}) \) is the initial configuration vector of SN P system \( \Pi \), where \( a_j^{(0)}, 1 \leq j \leq m \), represents the initial number of spikes in neuron \( \sigma_j \).
We say a rule $r_x \in R_j$, $1 \leq j \leq m$, of neuron $\sigma_j$ is applicable at time $k$, $k \geq 0$, if and only if the amount of spikes in the neuron, $a^{(k)}_j$, satisfies $E_x$ (or $a^{(k)}_j \in L(E_x)$). At some time $k$, if we have in neuron $\sigma_j$, $a^{(k)}_j \in L(E_x) \cap L(E_y)$, for some rules $r_x$, and $r_y$, $x \neq y$, then one of the two rules will be chosen to be applied. This is how the nondeterminism of the system is realized. Whereas several neurons with their chosen applicable rules can fire (or spike) simultaneously at time $k$, demonstrating parallelism.

A rule of type 2 from $R_j$ removes spike(s) from the neuron at some time $k$ when applied. Such a rule could only be applied if and only if the number of spikes in $\sigma_j$ is exactly the amount of spikes it needs to be applied. In particular, each rule $E/a^c \rightarrow a^p$; 0 of type (1) from $R_i$, if $a^s \rightarrow \lambda \in R_i$, then $a^s \notin L(E)$. Formally, if there exists a rule $a^s \rightarrow \lambda \in R_j$, then there cannot exist any rule $E/a^c \rightarrow a^p$ of type 1 in $R_i$ such that $a^s \in L(E)$.

In specifying rules in SNP systems, we follow the standard convention of simply not specifying $E$ whenever the left-hand side of the rule is equal to $E$.

The sequence of configurations defines a computation of the system. We say that a computation halts or reaches a halting configuration if it reaches a configuration where no more applicable rules are available. We could represent the output of the system either as the number of steps lapse between the first two spikes of the designated output neuron or as spike train. The sequence of spikes made by the system until the system reaches a halting configuration is called spike train.

For an SNP system, $\Pi$, with $m$ neurons, as introduced in Definition 1, we define below a matrix $M_\Pi$. In the sequel the rules of such an SNP system will be denoted $r_i$, $1 \leq i \leq n$, where $n$ is the total number of rules.

**Definition 3. (Spiking transition matrix)**

Let $\Pi$ be an SNP system with the total number of rules $n$ and $m$ neurons. The spiking transition matrix of $\Pi$ is $M_\Pi = [b_{ij}]_{n \times m}$, where

$$b_{ij} = \begin{cases} -c_i, & \text{if the left-hand side rule of } r_i \text{ in } \sigma_j \text{ is } a^c (c_i \text{ spikes are consumed}) \\ p_i, & \text{if the right-hand side of the rule } r_i \text{ in } \sigma_s \text{ is } a^p, (s \neq j \text{ and } (s, j) \in \text{syn}) \\ 0, & \text{otherwise} \end{cases}$$

The matrix $M_\Pi$ is (almost) a natural representation of the SNP system $\Pi$. Each row $i$, $1 \leq i \leq n$, corresponds to a rule $r_i : E_i/a^c \rightarrow a^p_i$ in some neuron $\sigma_j$, with $b_{ij}, 1 \leq j \leq m$, defined as above. Each column $j$, $1 \leq j \leq m$, corresponds to a neuron $\sigma_j$ and (i) for each rule belonging to $\sigma_j$, $r_i : E_i/a^c \rightarrow a^p_i$, for some $i, 1 \leq i \leq n$, $b_{ij} = -c_i$; (ii) for each rule $r_i : E_i/a^c \rightarrow a^p_i$ belonging to $\sigma_s$ for some $s, 1 \leq s \leq m$, such that $s \neq j$ and $(s, j) \in \text{syn}$, $b_{ij} = p_i$; (iii) otherwise, $b_{ij}$ equal to 0.

**Example 1.** An SNP system for $\mathbb{N} - \{1\}$.

Let $\Pi = ([a], \sigma_1, \sigma_2, \sigma_3, \text{syn, out})$, where $\sigma_1 = (2, R_1)$, with $R_1 = \{a^2/a \rightarrow$
$a, a^2 \rightarrow a$; $\sigma_2 = (1, R_2)$, with $R_2 = \{a \rightarrow a\}$; and $\sigma_3 = (1, R_3)$, with $R_3 = \{a \rightarrow a, a^2 \rightarrow \lambda\}$; $\text{syn} = \{(1, 2), (1, 3), (2, 1), (2, 3)\}$; $\text{out} = \sigma_3$.

\[ M_\Pi = \begin{pmatrix}
-1 & 1 & 1 \\
-2 & 1 & 1 \\
1 & -1 & 1 \\
0 & 0 & -1 \\
0 & 0 & -2
\end{pmatrix}. \]

Some more details regarding the properties of this matrix, $M_\Pi$, are presented in [1, 30].

The initial configuration of $\Pi$, introduced in Example \[1\] is $C_0 = (2, 1, 1)$.

**Definition 4. (Valid spiking vector)**

Let $C^{(k)} = (a_1^{(k)}, a_2^{(k)}, \ldots, a_m^{(k)})$, be the current configuration of an SN P system, $\Pi$, with a total of $n$ rules and $m$ neurons, at time $k$. Assume a total order is given for all $n$ rules of $\Pi$, so the rules can be referred to as $r_1, r_2, \ldots, r_n$. We denote a valid spiking vector by

$$Sp^{(k)} = (sp_1^{(k)}, sp_2^{(k)}, \ldots, sp_n^{(k)})$$

where

1. for each neuron $\sigma_j$, $1 \leq j \leq m$, if from all possible applicable rules of $R_j$ to $a_j^{(k)}$ at time $k$, a unique $r_i \in R_j$ is selected to be used, then $sp_i^{(k)} = 1$;

2. for all the rules $r_i$, $1 \leq i \leq n$, $r_i \in R_1 \cup \cdots \cup R_m$, that are not applicable at time $k$ or are not selected, if applicable, then $sp_i^{(k)} = 0$.

One can observe that $\sum_{i=1}^n sp_i^{(k)} \leq m$. 

Fig. 1. SN P system $\Pi$ without delay that generates the set $\mathbb{N} - \{1\}$.
We denote by \( \text{Sp}(0) = (\text{sp}_1(0), \text{sp}_2(0), \ldots, \text{sp}_n(0)) \), the initial valid spiking vector with respect to \( C^{(0)} \) of \( \Pi \). Note that \( \text{Sp}(0) \) need not be unique (see Example 1).

Definition 4 implies that \( \text{Sp}^{(k)} = (\text{sp}_1^{(k)}, \text{sp}_2^{(k)}, \ldots, \text{sp}_n^{(k)}) \) is a vector such that \( \text{sp}_i^{(k)} \in \{0, 1\} \), \( 1 \leq i \leq n \). The vector \( \text{Sp}^{(k)} \) indicates which rules must be fired (or used) at time \( k \) given the current configuration \( C^{(k)} \) of \( \Pi \). A vector is not a valid spiking vector if it does not satisfy Definition 4. In our example, the initial spiking vector with respect to \( C^{(0)} \) could be either \( \text{Sp}(0) = (1, 0, 1, 1, 0) \) or \( (0, 1, 1, 1, 0) \). The vector \( (1, 1, 1, 1, 0) \) is not a valid spiking vector.

Remark 1. (Net gain vector) [30]. The product \( \text{Sp}^{(k)} \cdot M_H \), denoted by \( \text{NG}^{(k)} \), represents the net amount of spikes the system obtained at time \( k \) from configuration \( C^{(k)} \). The sum \( \sum_{i=1}^m \text{sp}_i^{(k)} b_{ij} + \cdots + \text{sp}_n^{(k)} b_{nj} = 1 \leq i \leq n \), is the amount of spikes obtained in \( \sigma_j \) at time \( k \). Thus \( C^{(k+1)}(\sigma_j) = C^{(k)}(\sigma_j) + \text{sp}_1^{(k)} b_{1j} + \cdots + \text{sp}_n^{(k)} b_{nj} \), \( 1 \leq k \leq m \), where \( C^{(k)}(\sigma_j) \) and \( C^{(k+1)}(\sigma_j) \) are the amount of spikes in \( \sigma_j \) at time \( k \) and \( k+1 \), respectively.

Then we state the following important result from [1, 30]:

**Theorem 1. (Fundamental State Equation for SN P systems)**

Let \( \Pi \) be an SN P system with total of \( n \) rules and \( m \) neurons. Let \( C^{(k-1)} \) be given, then

\[
C^{(k)} = C^{(k-1)} + \text{Sp}^{(k-1)} \cdot M_H,
\]

where \( \text{Sp}^{(k-1)} \) is the valid spiking vector with respect to \( C^{(k-1)} \), and \( M_H \) is the spiking transition matrix of \( \Pi \).

**Definition 5. (Valid configuration)**

A configuration \( C \) of some SN P system \( \Pi \) is valid if and only if \( C = C^{(0)} \), or there exist valid configuration \( C' \) and valid spiking vector \( \text{Sp}' \), such that \( C = C' + \text{Sp}' \cdot M_H \), where \( M_H \) is the spiking transition matrix of SN P system \( \Pi \).

Using the formula from Theorem 1, we can obtain some valid configurations, \( C^{(k)} \), from the initial configuration.

**Corollary 1.** [1] Let \( \Pi \) be an SN P system with total of \( n \) rules and \( m \) neurons. Then at any time \( k \), we have

\[
C^{(k)} = C^{(0)} + \left( \sum_{i=0}^{k-1} \text{Sp}^{(i)} \right) \cdot M_H,
\]

where \( \text{Sp}^{(i)} \), \( 0 \leq i \leq k-1 \), is a valid spiking vector.

The configuration of SN P system \( \Pi \) is the state of \( \Pi \). This is the distribution of spikes among neurons in the system. We use Corollary 1 to define reachability of a configuration \( C \) of some SN P system \( \Pi \).
Definition 6. Let $M_{\Pi}$ be the spiking transition matrix of $\Pi$, and $C^{(0)}$ its initial configuration. A configuration $C$ is said to be $k$ reachable in $\Pi$ if and only if there is in $\Pi$ a sequence of $k$ valid configurations $\{C^{(i)}\}_{i=0}^{k}$, for some $k$, such that $C^{(i)} = C^{(i-1)} + Sp^{(i-1)} \cdot M_{\Pi}$, and $C^{(k)} = C$. Moreover, $C$ must be valid configuration.

We say the configuration $C$ is directly reachable from $C^{(k)}$, for some positive integer $k$ if and only if there exists $Sp^{(k)}$, such that $C = C^{(k)} + Sp^{(k)} \cdot M_{\Pi}$.

In particular, we define the set

$$R_{\Pi}(C^{(0)}) = \{C \mid C = C^{(0)} + \bar{S} \cdot M_{\Pi}, \text{where } \bar{S} \text{ is the sum of valid spiking vectors}\}$$

the set of all reachable configuration $C$ from $C^{(0)}$ in $\Pi$.

We note here that the vector $\bar{S}$ which is the sum of valid spiking vectors can be thought of as sequence of valid spiking vectors or valid spiking sequence.

3 Configuration graph $CG_{\Pi}$ of SN P system $\Pi$

Let $\Pi$ be an SN P system without delay. We describe a visual presentation of the computation of $\Pi$ as a directed graph.

Definition 7. Let $M_{\Pi}$ be the matrix representation of an SN P system $\Pi$. A configuration graph $CG_{\Pi}$ of $\Pi$ is the directed graph

$$CG_{\Pi} = (V, E),$$

where

$$V = \{C \mid C \in R_{\Pi}(C^{(0)})\}$$
$$E = \{(C^{(i)}, C^{(j)}) \mid C^{(j)} = C^{(i)} + Sp^{(i)} \cdot M_{\Pi}, \text{for some valid } Sp^{(i)}\}$$

The configuration graph $CG_{\Pi}$ of an SN P system $\Pi$ visually represents the (valid) computations of $\Pi$ as directed paths from $C^{(0)}$ to some node or configuration $C^{(k)}$, for some positive integer $k$ in $CG_{\Pi}$. One can think of $CG_{\Pi}$ as a computation tree of some SN P system $\Pi$.

Example 2. We provide in Fig. 2 the configuration graph $CG_{\Pi}$ of the SN P system $\Pi$ of Example 1.

Observation 1 Given a configuration graph $CG_{\Pi}$ of an SN P system $\Pi$, we have the following:

1. $(C^{(i)}, C^{(j)}) \in E(CG_{\Pi})$ if and only if $C^{(j)}$ is directly reachable from $C^{(i)}$.
2. $CG_{\Pi}$ is a labelled (directed) graph, where for every pair $C^{(i)}$ and $C^{(j)}$ of vertices, $(C^{(i)}, C^{(j)})$ is labelled with the valid spiking vector $Sp^{(i)}$.
3. The configuration graph $CG_{\Pi}$ of $\Pi$ is a one-component or connected directed graph.
4. $CG_{\Pi}$ is finite if and only if $R_{\Pi}(C^{(0)})$ is finite. Otherwise, $CG_{\Pi}$ is infinite.
Definition 8. Let $\Pi$ be an SN P system. Then the configuration graph $CG_\Pi$ of $\Pi$ is strongly connected if and only if for any two configurations $C, C' \in V(CG_\Pi)$, $C \in R_\Pi(C')$ and $C' \in R_\Pi(C)$.

Observation 2 Let $CG_\Pi$ be a strongly connected configuration graph of an SN P system $\Pi$. Then $CG_\Pi$ has a directed cycle.

4 Properties of SN P systems

We define below behavioral properties of SN P systems.

Definition 9. We call a rule $r$ live for an initial configuration $C^{(0)}$ if for every $C^{(k)}, C^{(k)} \in R_\Pi(C^{(0)})$, there exists a valid spiking sequence from $C^{(k)}$ that contains and applies $r$.

An SN P system $\Pi$ is live for $C^{(0)}$ if all its rules are live for $C^{(0)}$.

This means that all rules in the SN P system $\Pi$ must not be useless permanently during the computation. This is certainly not always the case of SN P systems especially, if an SN P system is generating or recognizing languages where $\Pi$ must halt, which means no rules can be applied anymore. This is equivalent to a so-called deadlock instance, that we can define as follows:

Definition 10. A deadlock in SN P system is a configuration where no rules can be applied.

An SN P system is deadlock-free for an initial configuration $C^{(0)}$, if no $C \in R_\Pi(C^{(0)})$ is a deadlock.

We relax our definition of liveness for SN P systems and introduce the case of a so-called non-dead or quasi-live rule in SN P system.

Definition 11. A rule $r$ is quasi-live for an initial configuration $C^{(0)}$, if there is a valid spiking sequence from $C^{(0)}$ that contains and applies $r$.

We call an SN P system quasi-live if all its rules are quasi-live.

Definition[11] of SN P systems liveness provides us allowance on some rules in the systems that may not be applied anymore after some time during the computation.

Most, if not all SN P systems so far constructed and reported in the literature are not deadlock-free. Certainly, one could provide the system with a set of neurons that supply the system with spikes to avoid “deadlock”. Note that if $\Pi$ has a “feedback-loop” mechanism then there is a possibility that $\Pi$ would be “live”.

Next, we define a bounded SN P system with respect to the amount of spikes every neuron has after computation. First, we define boundedness of a neuron.
Definition 12. A neuron $\sigma$ is **bounded** for an initial configuration $C^{(0)}$ if there is a positive integer $s$ such that, $C(\sigma) \leq s$, for every configuration $C \in R_{\Pi}(C^{(0)})$, where $C(\sigma)$ is the amount of spikes in $\sigma$ in configuration $C$. We say, $\sigma$ is an $s$-bounded neuron.

An SN P system $\Pi$ is **bounded** for an initial configuration $C^{(0)}$, if all neurons are bounded for $C^{(0)}$. $\Pi$ is $s$-**bounded** if all the neurons are $s$-bounded.

If $s = 1$, then we call the SN P system $\Pi$, **safe**.

Definition 13. An SN P system $\Pi$ is **reversible** if $C^{(0)} \in R_{\Pi}(C^{(k)})$, for any $k \geq 0$, where $C^{(k)} \in R_{\Pi}(C^{(0)})$.

Now we define below properties of SN P system that do not depend on a fixed initial configuration. These are called **structural properties** of SN P systems.

Definition 14. An SN P system $\Pi$ is **structurally live**, if there is an initial configuration $C^{(0)}$ such that $\Pi$ is live.

Definition 15. An SN P system $\Pi$ is **structurally bounded**, if it is bounded for any initial configuration $C^{(0)}$.

Definition 16. An SN P system $\Pi$ is **conservative**, if there exists an $n$-vector $\overline{y}$ of positive integers such that for every initial configuration $C^{(0)}$ and for every configuration $C \in R_{\Pi}(C^{(0)})$, we have $C \cdot \overline{y} = C^{(0)} \cdot \overline{y} = \text{a constant.}$

If the $n$-vector $\overline{y}$ is allowed to be non-negative vector of integers, then $\Pi$ is called **partial conservative**.

5 SN P systems and its configuration graph

In this section, we shall consider properties of SN P systems that depends on the systems configurations (behavioral properties) and those that depends only on the connectivity of the neurons (structural properties). We provide characterizations of these properties of SN P system $\Pi$ using it configuration graph $CG_{\Pi}$.

5.1 Some behavioral properties

Theorem 2. Let $\Pi$ be an SN P systems without delay. $\Pi$ is bounded if and only if $CG_{\Pi}$ is finite.

Proof. Let $\Pi$ be an SN P system without delay with $m$ neurons. The proof will have two parts.

($\Rightarrow$) Let $\Pi$ be bounded, then there is a non-negative integer $s$ such that $C(\sigma) \leq s$, for all neuron $\sigma$ in $\Pi$. This means that each neuron $\sigma$ can have spikes equal to either $C(\sigma) = 0, C(\sigma) = 1, \ldots,$ or $C(\sigma) = s$ during computations. Thus, we have only $(s + 1)^m$ possible reachable configurations in $CG_{\Pi}$, since $\Pi$ has $m$ neurons. Therefore, $CG_{\Pi}$ is finite.
identify a labelled directed path from $C$ to a sequence of valid spiking vectors that enables all the rules of $\Pi$. Since the amount of spikes in each neuron increases by at most $p$ spikes for each spiking sequence, where $p$ is the maximum spikes produced by any rule in $\Pi$. Then for every $\sigma$ in $\Pi$, $C(\sigma)$ is bounded above by $C(0)(\sigma) + p(k - 1)$. Therefore, $\Pi$ is bounded.

$\blacksquare$

**Theorem 3.** Let $\Pi$ be an SN P system without delay. $\Pi$ is deadlock-free if and only if all of the vertices in $CG_{II}$ have at least one synapse going out.

**Proof.** Let $\Pi$ be an SN P system without delay. We proceed as follows:

$(\implies)$ Suppose there is a $C \in V(CG_{II})$. Let us call the number of synapse going out of a node or configuration in $CG_{II}$ an $outdeg(C)$. Let $outdeg(C) = 0$. This means that after reaching $C$ from $C(0)$ the amount of spikes in each neuron at configuration $C$ is not enough to fire any of the rules in every neuron, and therefore reached a deadlock. This a contradiction, since $\Pi$ is deadlock-free. Therefore all vertices or configurations $C$ in $CG_{II}$ has $outdeg(C) \geq 1$.

$(\impliedby)$ Since each configuration $C$ in $CG_{II}$ has $outdeg(C) \geq 1$, then clearly that every configuration reached allows to have a valid spiking vector to transition from one configuration to the next and so on. Hence, $\Pi$ never reached a configuration where its computation stops or no more rule to apply. Therefore, $\Pi$ is deadlock-free.

$\blacksquare$

**Theorem 4.** Let $\Pi$ be an SN P system without delay. $\Pi$ is live if and only if for each vertex $C$ of $CG_{II}$, there exists a path from $C$ to $C^{(k)}$, for some positive integer $k$, such that the labels of the edges of the path indicate that they contain and apply all the rules of $\Pi$.

**Proof.** Let $\Pi$ be a live SN P system without delay.

$(\implies)$ Let $CG_{II}$ be the configuration graph of $\Pi$. Note that vertices of $CG_{II}$ are all reachable configurations of $\Pi$. Let $C = C^{(i)}$, for some integer $i$ such that $C^{(i)} \in R_{II}(C^{(0)})$. Let $r_1, r_2, r_3, \ldots, r_n$ be all the rules in $\Pi$. Since $\Pi$ is live, we can find a configuration from $C^{(i)}$ that enables at least a rule among the $n$ rules of $\Pi$. Let $r_1$ be applicable with respect to $C^{(i)}$. Then $C^{(i+1)} = C^{(i)} + Sp^{(i)} \cdot M_{II}$, for some valid spiking vector $Sp^{(i)}$ that enables $r_1$. With respect to $C^{(i+1)}$, let $Sp^{(i+1)}$ be valid spiking vector that enables, say, at least $r_2$. Then we have $C^{(i+2)} = C^{(i+1)} + Sp^{(i+1)} \cdot M_{II}$, and so on. This repeated applications of valid spiking vectors obtained from every succeeding configuration can be done due to liveness property of $\Pi$. This application will eventually lead us to a sequence of valid spiking vectors that enables all the rules of $\Pi$. Then finally reach some configuration $C^{(k)}$, for some positive integer $k$, such that $C^{(k)} = C^{(k-1)} + Sp^{(k-1)} \cdot M_{II}$ where $C^{(k)} \in R_{II}(C^{(i)})$. It can be realized that $C^{(k)} = C^{(i)} + (Sp^{(i)} + Sp^{(i+1)} + \cdots + Sp^{(k-1)}) \cdot M_{II}$. And from $CG_{II}$, we can identify a labelled directed path from $C^{(i)}$ to $C^{(k)}$, labelled sequentially by the
following valid spiking vectors \( S_p(i) \), \( S_p(i+1) \), \ldots, and \( S_p(k-1) \), which enable and apply all the \( n \) rules in \( \Pi \).

\( \iff \) All vertices of \( CG_H \) represents the reachable configurations of \( \Pi \) from \( C(0) \). Suppose \( C(i) \) is a vertex of \( CG_H \), for some positive integer \( i \), such that we can follow a labelled directed path from \( C(i) \) to a vertex \( C(k) \) for some positive integer \( k \). The labels are valid spiking vectors that indicate applicable rules of \( \Pi \), for every succeeding configuration. This implies \( C(k) = C(i) + \overline{\tau} \cdot M_H \), where \( \overline{\tau} \) is the sum of sequence of valid spiking vectors obtained starting from \( C(i) \) up to \( C(k-1) \). Moreover, all entries in \( \overline{\tau} \) are non-zero. Therefore, all rule of \( \Pi \) are applied in \( \overline{\tau} \), that implies liveness of \( \Pi \).

\( \blacksquare \)

**Theorem 5.** Let \( \Pi \) be an SN P system without delay. Then \( \Pi \) is reversible if and only if \( CG_H \) is strongly connected.

**Proof.** Let \( \Pi \) be an SN P system without delay.

\( \implies \) Suppose \( \Pi \) is reversible. Then for any positive integer \( k \), all reachable configuration \( C(k) \) from \( C(0) \), have directed path leading to \( C(0) \) from \( C(k) \). Suppose for any positive integer \( k' \), \( C(k') \in R_H(C(0)) \), such that \( k' \neq k \). Since \( \Pi \) is reversible, \( C(0) \in R_H(C(k')) \). Thus we can find a directed path in \( CG_H \) from \( C(k') \) to \( C(k) \). One can take the path drawn by the fact that \( C(0) \in R_H(C(k')) \), followed by the path from \( C(0) \) to \( C(k') \). Hence, for any positive integers \( k \) and \( k' \), we can find directed path from \( C(k) \) to \( C(k') \) and back. Therefore, \( CG_H \) is strongly connected.

\( \iff \) Let \( CG_H \) be strongly connected. Then \( C(0) \in R_H(\overline{C(k)}) \), for any positive integer \( k \). Therefore, \( \Pi \) is reversible.

\( \blacksquare \)

### 5.2 Some structural properties

We now look at some structural properties of SN P system \( \Pi \). These are properties of \( \Pi \) that are not dependent on \( C(0) \).

**Theorem 6.** Every live SN P system \( \Pi \) is structurally live.

**Proof.** This follows from the definition.

\( \blacksquare \)

To prove Theorem 7 below (by contradiction), we will use the result called Minkowski-Farkas Lemma \([11][14]\): Let \( A \) be an \( n \times m \) matrix over \( \mathbb{R} \). \( \overline{\tau} \) is a real column vector of size \( n \) and \( \overline{b} \) is a real column vector of size \( m \). The linear system \( A\overline{\tau} = \overline{b}, \overline{\tau} \geq 0 \) has a solution if and only if for all \( \overline{\tau} \in \mathbb{R}^m \), \( A^T \cdot \overline{\tau} \geq 0 \), we have \( \overline{\tau} \geq 0 \) and \( \overline{b} \geq 0 \).

**Theorem 7.** An SN P system \( \Pi \) without delay is structurally bounded if and only if there exists a positive column vector \( \overline{\tau} \) of integers of size \( n \) such that \( M_H \cdot \overline{\tau} \leq 0 \).
Proof. Let $\Pi$ be an SN P system without delay.

$(\Rightarrow)$ Suppose we have a positive column vector $\overline{y}$ of integers of size $n$, such that $M_{H} \cdot \overline{y} > 0$. Then by Minkowski-Farkas Lemma, there exists a row vector $\overline{s} \geq 0$ of integers of size $m$, such that $\overline{s} \cdot M_{H} > 0$. Then we can find configurations $C$ and $C'(0)$, such that $C - C'(0) = \overline{s} \cdot M_{H} > 0$ or $C > C'(0)$. We choose $C'(0)$ large enough to ensure a sequence of valid spiking vectors, $\overline{s}$, such that $\overline{s} = \overline{s}$ can be repeated indefinitely. Thus $\Pi$ is unbounded.

$(\Leftarrow)$ Suppose there exists a positive column vector $\overline{y}$ of integers of size $n$, such that $M_{H} \cdot \overline{y} \leq 0$. Let $C \in R_{H}(C'(0))$, where $C'(0)$ is some initial configuration of $\Pi$. This implies that $C = C'(0) + \overline{s} \cdot M_{H}$, where $\overline{s} \geq 0$. Then we obtain the following sum of products

$$C \cdot \overline{y} = C'(0) \cdot \overline{y} + \overline{s} \cdot M_{H} \cdot \overline{y}.$$ 

Since $M_{H} \cdot \overline{y} \leq 0$ and $\overline{s} \geq 0$,

$$C \cdot \overline{y} \leq C'(0) \cdot \overline{y}.$$ 

Let $C(\sigma_{i})$ be the number of spikes in $\sigma_{i}$ at configuration $C$ for some neuron $\sigma_{i}$ in $\Pi$. Let $\overline{y}(\sigma_{i})$ be the $i^{th}$ entry of $\overline{y}$. Then

$$C(\sigma_{i}) \leq \frac{C'(0) \cdot \overline{y}}{\overline{y}(\sigma_{i})}.$$ 

Thus, $C(\sigma_{i})$ is bounded for each $\sigma$.

\[\blacksquare\]

Theorem 8. An SN P system $\Pi$ without delay is conservative if and only if there exists a positive column vector $\overline{y}$ of integers of size $n$, such that $M_{H} \cdot \overline{y} = 0$.

Proof. Let $\Pi$ SN P system without delay.

$(\Rightarrow)$ Let $\Pi$ be conservative. Then by definition, there exists a positive column vector $\overline{y}$ of integers of size $n$, such that for every initial configuration $C'(0)$ and for every configuration $C \in R_{H}(C'(0))$, we have $C \cdot \overline{y} = C'(0) \cdot \overline{y}$ is a constant. Notice that $C \cdot \overline{y} = C'(0) \cdot \overline{y} + \overline{s} \cdot M_{H} \cdot \overline{y}$, for any positive column vector $\overline{y}$ of integers of size $n$. Since $\Pi$ is conservative, therefore,

$$M_{H} \cdot \overline{y} = 0,$$

since $\overline{s} \geq 0$.

$(\Leftarrow)$ Suppose we have a positive column vector $\overline{y}$ of integers of size $n$, such that $M_{H} \cdot \overline{y} = 0$. Then for every $C'(0)$ and every configuration $C \in R_{H}(C'(0))$, we have,

$$C \cdot \overline{y} = C'(0) \cdot \overline{y},$$ 

as required. Therefore, $\Pi$ is conservative.
Corollary 2. An SN P system $\Pi$ without delay is partial conservative if and only if there exists a positive column vector $\mathbf{y}$ of integers of size $n$, such that

$$M_{\Pi} \cdot \mathbf{y} = 0.$$ 

6 Final Remarks

We have demonstrated that the properties of SN P system without delay mimic the behavioral and structural properties of a (place/transition) Petri net [8, 23, 29]. We believe that several other properties of Petri nets could be interpreted as properties of SN P systems (see [15]). It is not hard to see that the converse of the definitions of the structural properties of SN P system as defined in Section [4] may not always be true.

The matrix representation defined in [1, 30] could be used to find other possible algebraic properties of SN P systems. Note when computing backwards [16] the fundamental state equation (1) is found useful. Solutions to the fundamental state equation for SN P system connote reachability of configurations.

Since SN P systems and their configuration graphs are directed graphs or digraphs, we could investigate further properties of these digraphs and perhaps we can observe, if any, their implications to the behavioral and structural properties of SN P systems. The techniques from [13, 24] on digraphs could be useful in possibly proving some properties of SN P systems using matrices. A good reference on the results on digraphs could be [2].

Also, we have demonstrated that strong connectedness is related to reversibility of $\Pi$. This may be relevant to periodicity and other possible behavioral and structural invariance in $\Pi$.

Finally, we suggest to explore further the behavioral and structural properties not only of SN P systems and variants but also the other types of P systems and their variants. We could either use the computation tree s in [16] or adopt our configuration graph for various types of P systems.

Acknowledgement

The author would like to thank the support from DOST-ERDT research grants; Semirara Mining Corp. Professorial Chair for Computer Science of College of Engineering, UPDiliman; RLC grant from UP-OVCRD.

References

1. Adorna, H.N., Matrix Representation of SN P systems: Revisited In: Gh. Păun, Proceedings of the 20th International Conference on Membrane Computing, Editura BIBLIOSTAR (2019) pp. 227-248.
2. Bang-Jensen, J., Gutin, G. Z., Digraphs Theory, Algorithms and Applications Springer-Verlag London Limited (2009).
3. Cabarle F.G.C., Adorna, H.N., On the Structures and behaviors of Spiking Neural P Systems and Petri Nets. In: E. Csuha-I-Vari ’u et al (eds): CMC 2012, LNCS 7762, (2013) pp 145-160.

4. Carandang, J.P., Villafores, J.M., Cabarle, F.G., Adorna, H.N., Martinez-del-Amor, M.A.. CuSNP: Spiking Neural P Systems Simulators in CUDA. Romanian Journal of Information Science and technology. Vol. 20, N. 1, (2017) pp 57-70

5. Chen, H., Ionescu, M., Ishdorj T-O., Păun, A, Păun, Gh., Perez-Jimenez, M.J. Spiking neural P systems with extended rules: universality and languages. Nat. Comput 7: (2008) pp 147 - 166.

6. Ciobanu, G., Păun, Gh., Pérez-Jiménez, M. J. Applications of Membrane Computing. Springer (2006).

7. Cordon-Franco, A., Gutierrez-Naranjo, M.A., Perez-Jimenez, M.J., Riscos-Nunez, A. Exploring computation trees associated with P systems. In: G. Mauri et al. (Eds) WMCS 2004. LNCS 3365, (2005) pp 278-286.

8. David, R., Alla, H.. Discrete, Continuous, and Hybrid Petri Nets, 2ed. Springer-Verlag Berlin Heidelberg (2010)

9. Dela Cruz, R.T.A., Jimenez, Z., Cabarle, F.G.C. , Adorna, H.N., Buño, K., Hernandez, N.H., and Carandang, J.P. Matrix Representation of Spiking Neural P Systems with Structural Plasticity, In: P.L. Fernandez, Jr., H.N. Adorna, A.A. Sioson, J.D.L. Caro (eds) Proc. 18th Philippine Computing Science Congress (PCSC2018), (2018) pp. 152-164.

10. Dela Cruz, R.T.A., Cailipan, D.P., Cabarle,F.G.C., Hernandez, N.H., Buño,K., Adorna, H.N, and Carandang, J.P. Matrix Representation and Simulation Algorithm for Spiking Neural P Systems with Rules on Synapses, In: P.L. Fernandez, Jr., H.N. Adorna, A.A. Sioson, J.D.L. Caro (eds) Proc. 18th Philippine Computing Science Congress (PCSC2018), (2018) pp.104-112.

11. Farkas, J.. Theorie der Einfachen Ungleichungne. Journal für die Reine und Angewandte Mathematik. 124, (1902). pp 1 -27.

12. Frisco, P., Gheorghe, M., Pérez-Jiménez, M. J.. Applications of Membrane Computing in Systems and Synthetic Biology. Springer (2014).

13. Frost, R.B.. Directed graphs and their adjacency matrices: Misconceptions and more efficient methods. Engineering Optimization. (1992). pp 225 -239.

14. Gale, D., Kuhn, H., Tucker, A.W.. Linear programming and the Theory of Games. In: Koopmans (eds) Activity Analysis of production and Allocation (1951). pp 317-329.

15. Gheorghe, M., Leficaru, R., Konur S., Nicolescu, I.M., Adorna, H.N., Spiking Neural P systems - Matrix representation and Formal verification. (preprint) (2020).

16. Gutierrez-Naranjo, M.A., Perez-Jimenez, M.J., Searching Previous Configurations in Membrane Computing, In: Paun et al (eds) WMC 2009. LNCS 5957, (2010) pp 301-315.

17. Ibarra, O.H., Perez-Jimenez, M.J., Yokomori, T.. On spiking neural P systems. Nat. Comput 9: (2010) pp 475-491

18. G.N. Ibo, H. Adorna; Characterizing Periodicity as a Dynamical Aspect of Generative SNP Systems , In: Verlan et al. Pre-Proceedings of Twelfth International Conference on Membrane Computing (CMC12) (2011) pp 225 -240

19. Ionescu, M., Paun, G., Yokomori, T.. Spiking Nueral P Systems. Fundamenta Informaticae 71 (2-3) (2006) pp 279-308.

20. Jimenez, Z.B. , Cabarle, F.G.C., de la Cruz, R.T.A., Buño,K.C., Adorna, H.N , Hernandez, N.H.S., and Zeng, X.. Matrix Representation and Simulation Algorithm of Spiking Neural P Systems with Structural Plasticity. Pre-Proc. Asian Conf. Membrane Computing (ACMC2018);(2018) pp 10-14
21. Juayong, R. Adorna, H.. Matrix Representation for Computations on Evolution-Communication P systems with Energy. In: Proceedings of 11th Philippine Computing Science Congress. (2011)

22. Martínez-del-Amor M.Á., Orellana-Martín, D. Cabarle, F.G.C. Pérez-Jiménez, M.J. Adorna, H.N.. Sparse-matrix Representation of Spiking Neural P Systems for GPU. Fifteenth Brainstorming Week on Membrane Computing (BWMC2017) Fénix Editora. Sevilla, Spain (2017) pp. 161–170.

23. Murata, T. Petri nets: properties, analysis and applications. Proceedings of the IEEE, vol.77 No. 4 (1989) pp 541-580.

24. Norman, R.L.. A Matrix Method for Location of Cycles of a Directed Graph. A.I.Ch.E. (1965) pp 450-452.

25. Pan, L., Wu, T., Zhang, Z.. A Bibliography of Spiking Neural P Systems. BULLETIN of the International Membrane Computing Society (I M C S) Vo. 1 No. 1 (2016) pp 63-78.

26. Păun, G.. Computing with Membranes. Journal of Computer System Sciences. 61 (1), (2000) pp 108 - 143

27. Păun, G. Membrane Computing - An Introduction. Springer, Berlin (2002).

28. Păun, G. Rozenberg, G., Salomaa, A. (eds): Handbook of Membrane Computing. Oxford University Press, Oxford, (2010)

29. Reisig, W., Rozenberg, G. Lectures on Petri Nets I: Basic Models, Advances in Petri Nets Springer-Verlag Berlin Heidelberg (1998)

30. Zeng, X., Adorna, H. Martínez-del Amor, M.A., Pan, L., Perez-Jimenez, M.J.. Matrix representation of Spiking Neural P Systems. In M. Gheorghe et al. (Eds): CMC 2010, LNCS 6510, (2010) pp 377-391

31. Zhang, G., Pérez-Jiménez, M. J., Gheorghe, M. Real-life Applications with Membrane Computing. Springer (2017).
Fig. 2. Configuration graph $CG_H$ of the SN P system $\Pi$ that generates the set $\mathbb{N} - \{1\}$. 