Arithmetic Operation in Spiking Neural P System with Chain Structure

Jing Luan *, Xiyu Liu
School of Management Science and Engineering,
Shandong Normal University,
Jinan, China
kaluanjing@163.com

Abstract: - Spiking neural P system with chain structure (SNPC, for short) is a new membrane system, which combines spiking neural P system (SNP, for short) with discrete Morse theory, that is to say, neural membrane cells in spiking neural P system are set on chain by discrete gradient vector path, building a SNP system with chain structure. Membrane computing is a computational model, which simulates the nature at the cellular level, and its maximum parallelism and well distributed manner showing strong computational completeness and efficiency. In this paper, the possibility of performing arithmetic operation with natural numbers in SNPC system has been proved, and the algorithms of the arithmetic operation in SNPC system are given from the aspects of generating datasets and generating string language. The efficiency and complexity of the algorithms are significantly improved, compared with usual computer architecture.
.

Key-Words: - Membrane Computing; Spiking Neural P System; Discrete Morse Theory; Arithmetic Operation

1 Introduction

Recently, natural computing, which simulates nature in the way it computes, has gone farther. Neural networks, Genetic algorithms and DNA Computing are the three fields which have been well established. However, natural computing is not only limited at the neural or genetic level, but at the cellular level. So we can imitate the computing process of biological system at a more specific level, such as a cell, to construct a computing model. Membrane computing (that is P system) is such model, which is a new branch of natural computing, and aimed at abstracting computing models from the structure and function of biological cells, and the collaboration among tissues, organs and other cell populations. P system is inspired by cell compartments and molecular membranes, and evolves in a distributed and maximally parallel manner. Recent results show that this model is a promising framework for solving NP-complete problems in polynomial time[1].

SNP system on chain is a new membrane structure, not network structure but chain structure, linking traditional membrane according to certain rules and forming a chain structure. The chain structure makes the process of membrane computing not select next reacting membrane at random, but react in accordance with the order of the membrane in the chain in successive. Avoid the time-consuming phenomenon resulted from randomness greatly and improve the computational efficiency of the SNP system. In this paper, based on the discrete Morse theory forming the SNP system with chain structure (SNPC, for short), the discrete gradient vector path is the standard to link the membranes, that is to say, along the discrete gradient vector path of the discrete Morse theory to form membrane structure. Morse theory had been proposed for a long time, and it is a useful tool in differential topology, used for investigating the topology of smooth manifolds, particularly for computer graphics, being the focus of the study. In the end of last century, Forman extended several aspects of the fundamental tool to discrete structures, providing an effective tool to describe the topology of the discrete objects, and made a big contribute to theory and applied mathematics. Its combinatorial aspect allows computation completely in- dependent of geometric realization, greatly enriching its range of applications. The literature [2] has proved the possibility of solving optimization problems with discrete Morse theory, showing the computational efficiency and power. The SNPC system, which combines SNP system with discrete Morse theory, enhances the computational power and efficiency of membrane system.

How to design a computational model of solving practical problems under a computational model framework is an important content of research. The maximum parallelism of membrane system makes it possible of completing arithmetic operation, and the complexity of the algorithm is lower. Addition, subtraction, division and multiplication (collectively called arithmetic operation) are regarded as basic operations of other complex processes, and are the oldest, most basic and initial parts of the mathematics. P system is a new computing model, and studying how to solve arithmetic operation by it can provide a theoretical basis for the realization of the system design of numerical calculation based on the P system. Literature [3] has proved the possibility of arithmetic operation by P system, and gives the algorithm for performing arithmetic operation of two arbitrary integers. Based on neurons sending spikes through synapses to other neurons, spiking neural P system is a new class of computing model of membrane computing, and regarding SNP system as a device for performing arithmetic operation starts from literature[4]. Now, there are SNP system for solving the sum of N natural numbers and SNP system for performing multiplication of two arbitrary natural numbers. Based on literature [3][4], in this paper, we proposed the arithmetic operation algorithms in SNPC system, which can solve arithmetic operations of two arbitrary natural numbers, and design two kinds of SNPC system from the aspects of generating datasets and generating string language. For the algorithms of this paper, the design of P system is easier, and the complexity of the computation is improved.

In this paper, the improved SNP system, that is to say, the SNP system with chain structure is introduced in section 2, and the formal presentation and rules are described. In section 3, the algorithm of performing arithmetic operation (addition, subtraction, division and multiplication) in SNPC system generating string language is given. The algorithm of performing arithmetic operation (addition and multiplication) in SNPC system generating datasets is proposed in section 4. In section 5, the compare of the two algorithms and the conclusion and outlook of SNPC system are introduced.

2 Spiking Neural P System with Chain structure

SNP system is a new computing device in the field of membrane computing. This new P system is proposed in 2006 by Lonescu et al. for the first time, which simulates the biological phenomena of collaborative of the neurons through synapses and processing spikes [5]. It is similar to the nervous system of the organism in the structure, and in the calculation it introduces the concept of the clock. Nerve spikes in the neural system triggered at a certain time interval, the neuron is inspired after receiving certain number of spikes, and transform the state to achieve the purposes of calculation.

Unlike the P system proposed before, SNP system has its own unique composition and operation mode: its basic calculating unit is neuron, and is indicated by the nodes in the directed graph commonly, the arc between nodes indicating synaptic. There are a certain number of a and a' in each neuron, which represent spike (a) or anti-spike (a'). The two are mutual antimatter, which counterbalance each other when encounter, that is to say, they don't co-exist [6] (in fact, there is rule $a'a \rightarrow \lambda$, which controls this process, and the rule $a'\ a \rightarrow \lambda$ doesn't appear obviously in general, and is implicit one which is with the highest priority, that is, it can be unlimited used until no reaction objects exit. However, in this paper, in order to show the function of anti-spikes, this rule $a'\ a \rightarrow \lambda$ is given obviously) . The rules in the SNP system are mainly two, that is to say, firing rules and forgotten rules. The firing rules make neuron send information to other neurons with spikes, and then other neurons can receive these spikes after a certain period of time. The forgotten rules are used to eliminate a certain number of spikes in the neurons.

P chain system is a new model of P system, which constructs membrane on the chain structure by some rules, getting P system with chain structure. Its outstanding advantage is that the communication between membranes is not at random, but in accordance with the order of the membrane in the chain in successive, successfully avoiding the time-consuming phenomenon caused by the random. In this paper, the rules of constructing the membrane structure we choose are combined with the discrete Morse theory, forming the membrane system over the discrete gradient vector path, that is to say, each cell on the discrete gradient vector path represents a membrane, and part or all of the path are considered as a membrane system. The discrete gradient vector path is a path satisfied[7]:

(1) A V-path is a sequence of simplices $\alpha^{(p)}$, $\beta^{(p+1)}$, $\alpha^{(p)}$, $\beta^{(p+1)}$, . . . , $\beta^{(p+1)}$, $\alpha^{(p)}$, such that for each i = (0, . . . , r + 1), {αi , βi } ∈ V and βi ≻ αi+1 ≠ αi ,

(2) if $r \geq 0$, $\alpha 0 = \alpha r{+}1$, such a path is a nontrivial closed path,

(3) a discrete vector field V on M is a collection of pairs $\{\alpha^{(p)}, \beta^{(p+1)}\}$ of simplicies of M with $\alpha \prec \beta$, such that each simplex is in at most one pair of V,
(4) a discrete vector field V is the gradient vector field of a discrete Morse function if and only if there are no nontrivial closed V-paths.

For SNPC system, which combines SNP system and P chain system, there are two main calculational modes in it, generating and accepting mode. In the generating mode, the membrane system is regarded as a generator, due to not need to input characters to the system, there is no need to specify the input neurons. For the accepting mode, in contrast to the generating mode, the SNPC system can omit the output neurons, and the data read from the environment through the input neurons. In the generating mode in SNPC system, computation starting from the initial configuration, there are two kinds of results, datasets and string language. For generating datasets, the outputs are defined as the time lag of output neuron exporting the first two spikes. Literature [5] has proved SNP system as a datasets generating device is complete in calculation, that is to say, being able to produce recursively enumerable collection of data. For generating string language, usually define a string corresponding with calculation in the following way, if at one step output neuron exports i spikes to environment, this step is correspond with string bi, where bi can be a new string or an empty string.

SNPC system is enrichment and expansion of P chain system and SNP system. It simulates the function of transmitting spikes of neurons and owns a chain structure. Similar to traditional SNP system, in SNPC system, there are single-alphabets, which represent spikes or anti-spikes; unlike to traditional SNP system, the link relationship among neurons is no longer network but partial order chain relationship; and in SNPC system, the regular expression and language in SNP system are generalized as restrains. The SNP system with chain structure of degree $m \geq 1$is a construct as the following form:

$\Pi = (O, \sigma 1, \sigma 2, \ldots, \sigma m, syn, in, out)$, where,

(1) 1) $O = \{a \text{ or } a'\}$ is the singleton alphabet (a is called spike and a' represents anti-spikes);

(2) $\sigma 1, \ldots, \sigma m$ are m neurons, $1 \leq i \leq m$, $\sigma 1, \sigma 2, \ldots, \sigma m$ are m cells abstracting from discrete gradient vector path, satisfied $\sigma 1 \prec \sigma 2 \prec \ldots \prec \sigma m$. Neurons σi is the form of $\sigma i = (ni, Ri)$, $1 \leq i \leq m$, where:

a. $ni \geq 0$ is the initial number of spikes contained by the neuron σi;

b. Ri is a finite set of rules of the following two forms:

1). firing rules: $C/a^c \rightarrow a^p$; d, $c \geq 1$, $p \geq 1$, $d \geq 0$, where C is the constraint condition of the rule, d is the delay time, that is, the interval between using the rule and releasing the spike, and $c \geq p$;

2). forgetting rules: $C'/a^s \rightarrow \lambda$, $s \geq 1$, where C' is the constraint condition of the rule, with the restriction that, for Ri in type 1) each rule $C/a^c \rightarrow a^p$; d, satisfies $C \cap C' = \phi$;

(3) $syn = \{1, 2, \ldots, m\}$ indicates the synapses between neurons, for any $1 \leq i \leq m$, there is $\sigma 1 \prec \sigma 2 \prec \ldots \prec \sigma m$;

(4) $in, out \in \{1, 2, \ldots, m\}$ indicate that the input and output neurons.

The rules 1) are used as followed: When restrain C is met, if there are k spikes in neuron σi and $k \geq c$, the rule $C/a^c \rightarrow a^p$; d, can be inspired in neuron σi. When this rule is used in neuron σi, it will consume c spikes (remaining $k - c$ spikes); at the same time, generate p spikes after d unit time, and immediately sent p spikes to its follow-up neuron. What's more, during the time from using this rule to transmitting the spikes generated, the neuron is closed, that is, don't receive any spikes. If the neuron fires the rule $C/a^c \rightarrow a^p$; d, $d \geq 1$ at the t step, the neuron is closed during the step $t, t + 1, \ldots, t + d - 1$. When a neuron is closed, all the rules over it can't be used, only if the neuron is open, the rules over it can be inspired. If a neuron sends spikes to a closed neuron, the closed neuron can't receive the spikes, and this spikes will disappear naturally. For the output neuron, it can send spikes to the environment.

The rules 2) are used as followed: When restrain C' is met, if there are k' spikes in neuron σi and $k' \geq s$, the forgetting rule $C'/a^s \rightarrow \lambda$, $s \geq 1$ can be inspired in neuron σi, and at this time all the firing rules and generating rules over the neuron can't be used. When this rule is used in neuron σi, it will consume s spikes, but don't generate new spikes.

There can be a lot of rules in a neuron. If there are many rules can be used at once in neuron σi, in this neuron, there are only one rule can be used, which is selected at random. The diagram of SNPC system as shown in Figure 1.

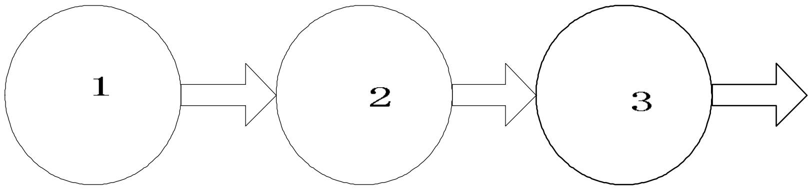

Fig. 1: The SNPC system

In Fig.1, there are three membranes in the chain structure membrane system, where membrane 1 is the input membrane, whose objects can be generated by the reaction in the membrane, or can be from the skin membrane, and membrane 3 is the output membrane, whose objects are sent to environment directly. The arrow represents synapse whose function is linking the neighbouring neurons.

3 Arithmetic Operation in SNPC System Generating String Language

The SNPC system generating string language is referred to define a string corresponding with calculation, if at one step output neuron exports i spikes to environment, this step is correspond with string bi , where bi can be a new string or an empty string. In this paper, we use i represents the outputs, that is to say, the number of spikes exported by the output neuron is seen as the results. Then the algorithm that performing arithmetic operation (addition, subtraction, division and multiplication) of two arbitrary natural numbers in SNPC system generating string language is given.

3.1 Addition

For achieving the addition of two arbitrary natural numbers m and n ($m \geq n$) in SNPC system generating string language, the structure of the membrane system is designed as followed:

$\prod(Addition) = (O, \sigma 1, \sigma 2, syn, in, out)$, where,

- $O = \{a\}$;
- $\sigma 1 = \{n1, R1\}$, $\sigma 2 = \{n2, R2\}$;
- $n1 = \{m|a\}$, $n2 = \{n|a\}$,
- $R1 = \{a^m \rightarrow a^m\}$,
  $R2 = \{a^m / \{a^m \rightarrow a^m\}, \{a^n \rightarrow a^n\}\}$;
- in=1, out=2.

In the whole P system, when all rules which can be inspired are applied once, we call it one step. In other words, in each step, all rules which can be applied have to be applied to all possible objects. As shown is Fig.2, in membrane $\sigma 1$, whose function is to generate m spikes a, there are m objects a and one firing rule $a^m \rightarrow a^m$. The m spikes a generated will be sent to membrane $\sigma 2$ at once, then the restrain a^m met, the rule $R2 = \{a^m / \{a^m \rightarrow a^m\}, \{a^n \rightarrow a^n\}\}$ in $\sigma 2$ is fired. The rule $R2$ is chained rule, which is applied as followed which refers to the literature [8]: there is a chained rules $R2$, which contains two vectors of rules. In the membrane, if the first rule of chained rules R is applied, then in the next step the rest of the rules from R will be applied in order in consecutive steps. However, if a rule from an already started vector of chained rules R can't be applied, then the execution of R is dropped, that is, for the current application of R, the remaining rules are not executed anymore. In SNPC system, if and only if a chained rule is finished, the spikes generated can be sent to its follow-up neuron. When $m > n$, at the beginning the restrain am can't be met, when receive the m spikes a from membrane $\sigma 1$, the firing rule $a^m \rightarrow a^m$ is executed firstly, producing m spikes a, then firing rule $a^n \rightarrow a^n$ is inspired, generating n spikes a. At present there are $m + n$ spikes a totally which will be sent to the environment. On the condition of $m = n$, at the beginning the restrain a^m in membrane $\sigma 2$ met, the chained rule $R2$ can execute, where rule vector $a^m \rightarrow a^m$ is inspired firstly , generating m spikes a, then rule vector $a^n \rightarrow a^n$ is fired, generating n spikes a sent to environment immediately. When receive the m spikes a from membrane $\sigma 2$, the chained rule $R2$ is fired again and sent n spikes a to environment. Until then, the SNPC system completes the addition ($m + n$) of two arbitrary natural numbers m and n ($m \geq n$).

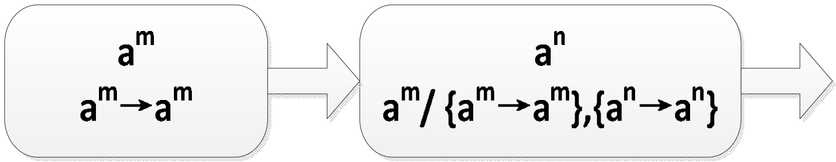

Fig. 2: Addition in SNPC system generating string language

3.2 Subtraction

For achieving the subtraction of two arbitrary natural numbers in SNPC system generating string language, the structure of the membrane system is designed as followed:

∏(*Subtraction*) = (*O, σ*1*, σ*2*, syn, in, out*), where,
- $O = \{a, a'\}$;
- $\sigma 1 = \{n1, R1\}$, $\sigma 2 = \{n2, R2\}$;
- $n1 = \{m|a\}$, $n2 = \{n|a'\}$,
- $R1 = \{a^m \rightarrow a^m\}$, $R2 = \{a^m / aa' \rightarrow \lambda, a \rightarrow a\}$;
- in=1, out=2.

As shown is Fig.3, in membrane $\sigma 1$, whose function is to generate m spikes a, there are m objects a and one firing rule $a^m \rightarrow a^m$. The m spikes a generated will be sent to membrane $\sigma 2$ at once, then the rule $R2 = \{a^m/aa' \rightarrow \lambda, a \rightarrow a\}$ in $\sigma 2$ is fired. Due to high priority, the rule $aa' \rightarrow \lambda$ is executed firstly, spike a counterbalanced by anti-spike a' until n anti-spikes a' is run out, then firing rule $a \rightarrow a$ is inspired, the remaining $(m - n)$ spikes a from membrane $\sigma 1$ sent to environment. At present there are $(m - n)$ spikes a totally which will be sent to the environment. Until then, the SNPC system completes the subtraction $(m - n)$ of two arbitrary natural numbers m and n.

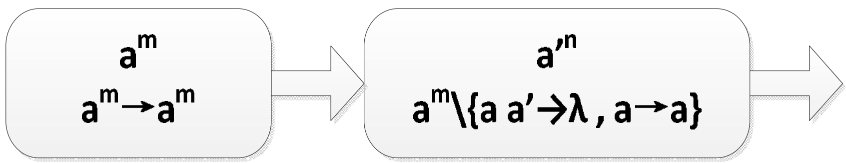

Fig. 3: Subtraction in SNPC system generating string language

3.3 Multiplication

For achieving the multiplication of two arbitrary natural numbers n and m $(n \geq m)$ in SNPC system generating string language, the structure of the membrane system is designed as followed:

∏(*multiplication*) = (*O, σ*1*, σ*2*, syn, in, out*), where,

- $O = \{a, a'\}$;
- $\sigma 1 = \{n1, R1\}$, $\sigma 2 = \{n2, R2\}$;
- $n1 = \{n|a'\}$, $n2 = \{(n+1)^2|a\}$,
- $R1 = \{a' \rightarrow a'\}$,
  $R2 = \{a'/\{aa' \rightarrow \lambda\}, \{a^n \rightarrow a^m\}\}$;
- in=1, out=2.

As shown is Fig.4, in membrane $\sigma 1$, whose function is to generate spikes a', there are n objects a' and one firing rule $a' \rightarrow a'$. The spikes a' generated will be sent to membrane $\sigma 2$ at once, then the restrain a' met, the rule $R2 = \{a'/\{aa' \rightarrow \lambda\}, \{a^n \rightarrow a^m\}\}$ in $\sigma 2$ is fired, where the rule $R2$ is chained rule too. Due to high priority, the rule $aa' \rightarrow \lambda$ is executed firstly, spike a counterbalanced by anti-spike a', then with the existence of $(n + 1)^2$ object a, firing rule $a^n \rightarrow a^m$ is inspired, generating m spike a. Because of the existence of object a', the rule $R1$ can execute until the anti-spike a' doesn't exist. Along with the anti-spike a' sent to membraneσ2 from membrane σ1 the chained rule *R2* is red again until the anti-spike a' or the object α is run out. The specific execution is as shown in Table 1.

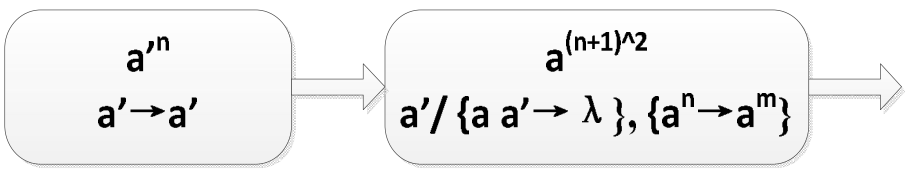

Fig. 4: Multiplication in SNPC system generating string language

Table1 Process of multiplication in SNPC system generating string language

Step	Neuron1			Neuron2			output
	start		end	start		end	
1	a'^{n}	$a'\to a'$	$a'^{(n-1)}$	a' (1)		a'	
2	$a'^{(n-1)}$	$a'\to a'$	$a'^{(n-2)}$	a' $a^{((n+1)^2)}$	$a'/\{aa'\to\lambda\}$, $\{a^{n}\to a^{m}\}$	$a^{((n+1)^2-(n+1))}$	a^{m}
3	$a'^{(n-2)}$	$a'\to a'$	$a'^{(n-3)}$	a' $a^{((n+1)^2-(n+1))}$	$a'/\{aa'\to\lambda\}$, $\{a^{n}\to a^{m}\}$	$a^{((n+1)^2-2(n+1))}$	a^{2m}
4	$a'^{(n-3)}$	$a'\to a'$	$a'^{(n-4)}$	a' $a^{((n+1)^2-2(n+1))}$	$a'/\{aa'\to\lambda\}$, $\{a^{n}\to a^{m}\}$	$a^{((n+1)^2-3(n+1))}$	a^{3m}
i-1	$a'^{(n-(i-2))}$	$a'\to a'$	$a'^{(n-(i-1))}$	a' $a^{((n+1)^2-(n+1)(i-3))}$	$a'/\{aa'\to\lambda\}$, $\{a^{n}\to a^{m}\}$	$a^{((n+1)^2-(n+1)(i-2))}$	$a^{(i-2)m}$
i	$a'^{(n-(i-1))}$	$a'\to a'$	$a'^{(n-i)}$	a' $a^{((n+1)^2-(n+1)(i-2))}$	$a'/\{aa'\to\lambda\}$, $\{a^{n}\to a^{m}\}$	$a^{((n+1)^2-(n+1)(i-1))}$	$a^{(i-1)m}$
i+1	$a'^{(n-i)}$	$a'\to a'$	$a'^{(n-(i+1))}$	a' $a^{((n+1)^2-(n+1)(i+1))}$	$a'/\{aa'\to\lambda\}$, $\{a^{n}\to a^{m}\}$	$a^{((n+1)^2-(n+1)(i))}$	$a^{(i)m}$
n-1	$a'^{(2)}$	$a'\to a'$	a'	a' $a^{((n+1)^2-(n+1)(n-3))}$	$a'/\{aa'\to\lambda\}$, $\{a^{n}\to a^{m}\}$	$a^{((n+1)^2-(n+1)(n-2))}$	$a^{(n-2)m}$
n	a'	$a'\to a'$	φ	a' $a^{((n+1)^2-(n+1)(n-2))}$	$a'/\{aa'\to\lambda\}$, $\{a^{n}\to a^{m}\}$	$a^{((n+1)^2-(n+1)(n-1))}$	$a^{(n-1)m}$
n+1				a' $a^{((n+1)^2-(n+1)(n-1))}$	$a'/\{aa'\to\lambda\}$, $\{a^{n}\to a^{m}\}$	$a^{((n+1)^2-(n+1)(n))}$	$a^{(n)m}$

As shown in Table 1 the SNPC system completes the multiplication (m*n) of two arbitrary natural numbers m and *n* (n⩾ m)

3.4 Division

For achieving the division of two arbitrary natural numbers n and m ($n \geq m$) in SNPC system generating string language, the structure of the membrane system is designed as followed:

$\prod(Division) = (O, \sigma1, \sigma2, syn, in, out)$, where,

- $O = \{a, a'\}$;
- $\sigma1 = \{n1, R1\}$, $\sigma2 = \{n2, R2\}$;
- $n1 = \{n|a'\}$, $n2 = \{2n|a\}$,
- $R1 = \{a' \to a'\}$,
  $R2 = \{a'/\{aa' \to \lambda\}, \{a^{(m-1)} \to a\}\}$;
- in=1, out=2.

As shown is Fig.5, in membrane $\sigma1$, whose function is producing n a' spikes, there are n objects a' and one firing rule $a'\to a'$. The anti-spike a' generated will be sent to membrane $\sigma2$ at once, then the restrain a' met, the rule $R2 = \{a'/\{aa' \to \lambda\}, \{a^{(m-1)} \to a\}\}$ in $\sigma2$ is fired, where the rule $R2$ is chained rule too. Due to high priority, the rule $aa' \to \lambda$ is executed firstly, spike a counterbalanced by anti-spike a', then the firing rule $a^{(m-1)} \to a$ is executed, consuming $(m-1)$ spike a and generating one a spike. Because of the existence of object a', the rule $R1$ can execute until the anti-spike a' doesn't exist. Along with the anti-spike a' sent to membrane $\sigma2$ from membrane $\sigma1$, the chained rule $R2$ is fired again until the anti-spike a' or the object a is run out. The specific execution of aliquot operation is as shown in Table 2. For the case of aliquant operation, the process is similar with aliquot operation, except for the halting conditions–the number of object a less than m.

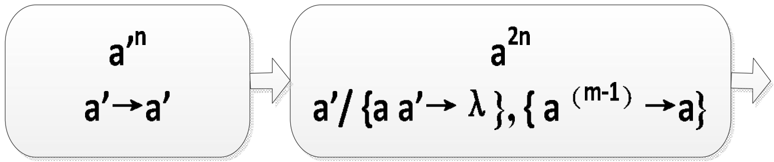

Fig. 5: Division in SNPC system generating string language

Table2 Process of division in SNPC system generating string language

Step	Neuron1			Neuron2			output
	start		end	start		end	
1	a'^{n}	$a' \to a'$	$a'^{(n-1)}$	a' (1)		a'	
2	$a'^{(n-1)}$	$a' \to a'$	$a'^{(n-2)}$	a' $a^{(2n)}$	$a'/\{a\,a' \to \lambda, a^{(m-1)} \to a\}$	$a^{(2n-m)}$	a
3	$a'^{(n-2)}$	$a' \to a'$	$a'^{(n-3)}$	a' $a^{(2n-m)}$	$a'/\{a\,a' \to \lambda, a^{(m-1)} \to a\}$	$a^{(2n-2m)}$	2a
4	$a'^{(n-3)}$	$a' \to a'$	$a'^{(n-4)}$	a' $a^{(2n-2m)}$	$a'/\{a\,a' \to \lambda, a^{(m-1)} \to a\}$	$a^{(2n-3m)}$	3a
i-1	$a'^{(n-(i-2))}$	$a' \to a'$	$a'^{(n-(i-1))}$	a' $a^{(2n-(i-3)m)}$	$a'/\{a\,a' \to \lambda, a^{(m-1)} \to a\}$	$a^{(2n-(i-2)m)}$	(i-2)a
i	$a'^{(n-(i-1))}$	$a' \to a'$	$a'^{(n-i)}$	a' $a^{(2n-(i-2)m)}$	$a'/\{a\,a' \to \lambda, a^{(m-1)} \to a\}$	$a^{(2n-(i-1)m)}$	(i-1)a
i+1	$a'^{(n-i)}$	$a' \to a'$	$a'^{(n-(i+1))}$	a' $a^{(2n-(i-1)m)}$	$a'/\{a\,a' \to \lambda, a^{(m-1)} \to a\}$	$a^{(2n-im)}$	ia
k-1	$a'^{(n-(k-2))}$	$a' \to a'$	$a'^{(n-(k-1))}$	a^{3m}	$a'/\{a\,a' \to \lambda, a^{(m-1)} \to a\}$	a^{2m}	(k-2)a
k	$a'^{(n-(k-1))}$	$a' \to a'$	$a'^{(n-k)}$	a^{2m}	$a'/\{a\,a' \to \lambda, a^{(m-1)} \to a\}$	a^{m}	(k-1)a
k+1	$a'^{(n-k)}$	$a' \to a'$	$a'^{(n-(k+1))}$	a^{m}	$a'/\{a\,a' \to \lambda, a^{(m-1)} \to a\}$	φ	ka

As shown in Table 2, the SNPC system completes the division (m/n) of two arbitrary natural numbers *m* and *n* ($n \geq m$).

4 Arithmetic Operation in SNPC System Generating Datasets

The SNPC system generating datasets is referred to the outputs are defined as the time lag of output neuron exporting the first two spikes. In this paper, the one step of the computing process is seen as a time unit, and the result is the steps of output neuron exporting the first two spikes. Due to the irreversibility of the time, the SNPC system generating datasets is more suitable for the addition and multiplication. Now the algorithm that performing arithmetic operation (addition and multiplication) of two arbitrary natural numbers in SNPC system generating datasets is given.

4.1 Addition

For achieving the addition of two arbitrary natural numbers in SNPC system generating datasets, the structure of the membrane system is designed as followed:

$\prod(Addition) = (O, \sigma1, \sigma2, \sigma3, syn, in, out)$, where,

- $O = \{a\}$;
- $\sigma1 = \{n1, R1\}$, $\sigma2 = \{n2, R2\}$, $\sigma3 = \{n3, R3\}$;
- $n1 = \{1|\alpha\}$, $n2 = \{O|\Phi\}$, $n3 = \{1|\alpha\}$;
- $R1=\{a/a \to a; m-1\}, R2=\{a \to a; n-1\}, R3=\{a \to a; 0\}$;
- in=l, out=3

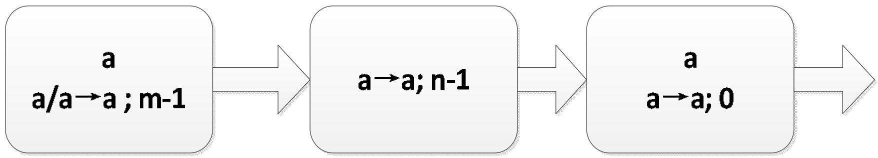

Fig. 6: Addition in S NPC system generating datasets

As shown in Fig.6, there are one object a and one firing rule a/a → a; m−1 (m−1 is the delay)in membrane σ1. There is object a in membrane σ1, the restrain a met, so the rule a/a→a; m−1 can be executed, consuming one a spike and generating one a spike. Because of the delay m−1, from the step 2 to step m, the membrane σ1 is closed, not performing the rules, receiving and sending spikes. In membrane σ2, due to the lack of object a, the rule a→a;n−1 can't be fired until receives the object a from membrane σ1. There are one object a and one firing rule a→a;0 in membrane σ3. Due to the existence of object a, at the step one, the rule a→a;0 can be inspired, and send one a spike to environment at once. This is the first time the system exports objects to environment. When receive the a spike from membrane σ2, the rule a→a;0 will be inspired again, and send one a spike to environment at once. This is the second time the system exports objects to environment. The step lag between the two exports is the computing result.
The specific execution is as shown in Table 3.

Table 3 The process of addition in SNPC system generating datasets

Step	Neuron1			Neuron2			Neuron3		
	start		end	start		end	start		end
1	a	a→a	a				a	a→a	φ
2	a	inactive	a						
3	a	inactive	a						
4	a	inactive	a						
m	a	inactive release a	φ	a		a			
m+1				a	a→a	a			
m+2				a	inactive	a			
m+3				a	inactive	a			
m+n				a	inactive release a	φ	a		a
m+n+1							a	a→a	φ

As shown in Table 3, the SNPC system completes the addition (m+n) of two arbitrary natural numbers *m* and *n*.

4.2 Multiplication

For achieving the multiplication of two arbitrary natural numbers in SNPC system generating datasets, the structure of the membrane system is designed as followed:

$\prod(Multiplication) = (O, \sigma 1, \sigma 2, \sigma 3, syn, in, out)$, where,

- $O = \{a, a'\}$;
- $\sigma 1 = \{n1, R1\}$, $\sigma 2 = \{n2, R2\}$, $\sigma 3 = \{n3, R3\}$;
- $n1 = \{m|a\}$, $n2 = \{0|\phi\}$, $n3 = \{1|a'\}$;
- $R1 = \{a/a \rightarrow a; n-2\}$, $R2 = \{a \rightarrow a; m\}$, $R3 = \{a' \rightarrow a', a^m \rightarrow a^m; 0\}$;
- in =1, out=3.

As shown in Fig.7, there are *m* objects *a* and one firing rule $a/a \rightarrow a$; $n-2$ ($n-2$ is the delay) in membrane $\sigma 1$. There is object *a* in membrane $\sigma 1$, the restrain *a* met, so the rule $a \rightarrow a$; $n-2$ can be executed, consuming one *a* spike and generating one *a* spike. Because of the delay $n - 2$, from the step 2 to step $n - 1$, the membrane $\sigma 1$ is closed, not performing the rules, receiving and sending spikes. In membrane $\sigma 2$, due to the lack of object *a*, the rule $a \rightarrow a$; *m* can't be fired until receives the object *a* from membrane $\sigma 1$. There are one object a' and one firing rule $a' \rightarrow a'$; 0 in membrane $\sigma 3$. Due to the existence of object a', at the step one, the rule $a' \rightarrow a'$; 0 can be inspired, and send one a' spike to environment at once. This is the first time the system exports objects to environment. When receive *m a* spikes from membrane $\sigma 2$, the rule $a^m \rightarrow a^m$; 0 will be inspired, and send *m a* spikes to environment at once. This is the second time the system exports objects to environment. The step lag between the two exports is the computing result.

For performing multiplication in SNPC system generating datasets, m=3, n=6 as example, the specific execution is as shown in Table 4.

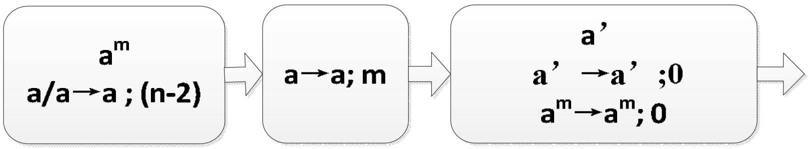

Fig. 7: Multiplication in SNPC system generating datasets

Table 4 The process of Multiplication in SNPC system generating datasets

Step	Neuron1			Neuron2			Neuron3		
	start		end	start		end	start		end
1	a^3	a→a	a^3				a’	a’→a’	φ
2	a^3	inactive	a^3						
3	a^3	inactive	a^3						
4	a^3	inactive	a^3						
5	a^3	inactive release a	a^2	a		a			
6	a^2	a→a	a^2	a	a→a	a			
7	a^2	inactive	a^2	a	inactive	a			
8	a^2	inactive	a^2	a	inactive	a			
9	a^2	inactive	a^2	a	inactive release a	φ	a		a
10	a^2	inactive release a	a	a		a			
11	a	a→a	a	a	a→a	a			
12	a	inactive	a	a	inactive	a			
13	a	inactive	a	a	inactive	a			
14	a	inactive	a	a	inactive release a	φ	a^2		a^2
15	a	inactive release a	φ	a		a			
16				a	a→a	a			
17				a	inactive	a			
18				a	inactive	a			
19				a	inactive release a	φ	a^3		a^3
20							a^3	$a^3 \rightarrow a^3$	φ

As shown in Table 4, the SNPC system completes the multiplication (m+n) of two arbitrary natural numbers *m* and *n*.

5 Conclusion

Due to the objects of each membrane in membrane system can evolve in the way of maximum parallel principle with high parallelism, P system, a computational model, is with the computing power to solve complex problems. From the above deduction process, we can see the SNPC system with high parallelism can solve arithmetic operation of any two natural numbers in polynomial time. Compared with the algorithm in P system, cell-like P system, proposed in literature [3], for the algorithms in this paper, the design of P system is easier, and the efficiency of the computation is improved. The two algorithms proposed in this paper, one based on generating string language, resulted as the number of strings exported, and the other based on generating datasets, resulted as the time lag between the first two exports. For addition, the time complexity of the two algorithms is linear, but the executive efficiency of SNPC system generating string language is higher. For multiplication, the time complexity of the algorithm in SNPC system generating string language is linear, but in SNPC system generating datasets is polynomial time, so the algorithm in SNPC system generating string language is easier and more efficient with high universality, which is suited for subtraction and division. For performing arithmetic operation in SNP system, how to import the natural numbers to the system is the question considered firstly, and the binary encoding is the frequently-used encoding method currently [4]. In this paper, the problem isn't considered when perform arithmetic operation in SNPC system which is the deficiencies of algorithm design and need to be improved. There are some other questions needed to be discussed, expect the design of reasonable hardware based on SNPC system, such as performing all possible arithmetic operation in SNPC system, more precise, when the objects aren't natural numbers, but negative numbers for example, how to design the SNPC system to solve the problem.

Membrane computing is considered to be the most promising in solving NP problems, of course, SNPC system also can used to solve NP problems. Literature [9] proposed solving TSP problem by membrane computing, using the algorithm MCTSP, and on this basis, we can use the SNPC system to find a more efficient algorithm for solving TSP. All of these are the future direction and points of the study. Obviously, due to the current of the analog implementation of the membrane system and in reality the corresponding technical support of realizing are too little, calculations based on membrane computing models and methods are more difficult to verify, which makes to solving the problem by membrane computing relatively vague. These have also become challenges and efforts for the study of SNPC system.

References:

[1] T. O., Ishdorj. "Membrane Computing, Neural Inspirations, Gene Assembly in Ciliates". Ph.D. Thesis, Universidad de Sevilla, 2006.
[2] Zhang Jian-ping, Liu Xi-yu. Optimization Model Based on Discrete Morse theory. System Engineering- Theory & Practice, vol.31, 2011.
[3] Ping Guo, Jing Chen. Arithmetic Operation in Membrane System. 2008 International Conference on BioMedical Engineering and Informatics.
[4] M.A. Gutirrez-Naranjo, A. Leporati. Performing arithmetic operations with spiking neural P systems[C]. Preproceedings of the 7th Brainstorming Week on Membrane Computing, 2009:181-198.
[5] Ionescu M, Paun Gh, Yokomori T. Spiking neural P systems. Fundamenta Informaticae, 2006, 71(2-3) : 279-308.
[6] Linqiang Pan, Gheorghe Paun. Spiking Neural P Systems with Anti-Spikes. Int. J. of Computers, Communications & Control, ISSN 1841-9836, E-ISSN 1841-9844 Vol. IV (2009), No. 3, pp. 273-282.
[7] Forman Robin. Users Guide to Discrete Morse Theory. S'eminaire Lotharingien de Combinatoire, Article B48c,vol.48, pp.1-35, 2002.
[8] Dragos Sburlan. P Systems with Chained Rules. Twelfth International Conference on Membrane Computing ,pp.447-458.
[9] Chen Hai-zhu. Application Research on Membrane Computing[D]. Chongqing: Chongqing Uni- versity, 2011.