Variants of Spiking Neural P Systems with Coloured Spikes

Rudolf FREUND

Sergiu IVANOV

Marion OSWALD

BWMC 2017

Extended Spiking Neural P System (ESNPS)

We consider several variants of extended spiking neural P systems (ESNPSs) allowing for spikes with different colors (ECSNPSs).

Instead of choosing the rules in the neurons based on the current contents being in a regular set, we also consider the way of choosing the rule to be applied in each neuron to consume the minimal energy. This choice can be accomplished by assigning energy values

- to each rule or
- directly to the colored spikes.

ECSNPS with Request (ECSNPRS)

We now also consider the variants of ECSNPSs with requesting spikes instead of sending spikes, i.e., ECSNPS with request (ECSNPRS).

In order to allow these systems to increase the number of spikes in the system, we use rules requesting spikes from the environment where all spikes are assumed to be available in an unbounded number.

Extended Spiking Neural P System (ESNPS)

An extended spiking neural P system (of degree $m \geq 1$) (an ESNPS for short) is a construct $\Pi = (N,I,R)$ where

- N is the set of cells (or neurons); uniquely identified by a number between 1 and m;
- I describes the initial configuration by assigning an initial value (of spikes);
- R is a finite set of *spiking rules* of the form $i{:}E/a^k \rightarrow P$, $i \in \{1,\ldots,m$, E is the *checking set*, a regular set over a, $k \in N$ is the "number of spikes" (the energy) consumed by this rule, and P is a (possibly empty) set of productions of the form (n,w_n), $n \in \{1,\ldots,m\}$, $w_n \in \{a\}^*$ is the weight of the energy sent along the axon from neuron i to neuron n.

Extended Spiking Neural P System (ESNPS)

A *configuration* of the ESNPS system Π is described by the actual number of spikes in each neuron.

A *transition* from one configuration to another one works as follows: for each neuron i, i∈{1,...,m}, we non-deterministically choose an applicable rule $i:E/a^k \rightarrow P$, i.e., the number of spikes in neuron i is in the regular set E; the application of this rule reduces the number of spikes in neuron i by k and adds w_n spikes to each neuron n specified in P.

The computation of Π *halts* if no spiking rule can be applied any more.

ESNPS with Colored Spikes (ECSNPS)

We may extend this model of ESNPSs by allowing more than one variant of spikes, i.e., we may use *colored* spikes a, b, etc.; the set of these colored spikes is denoted by U.

In that case, the spiking rules then are of the form $i{:}E/u \to P$ where E is a regular multiset over the finite set of colored spikes U, u is a finite multiset over U, and P is a (possibly empty) set of productions of the form (n,w_n) where $n \in \{1,...,m\}$ (thus specifying the target neuron) and w_n is a finite multiset of colored spikes over U sent along the axon from neuron i to neuron n.

ECSNPS with Energy Control

We consider two variants: the first variant assigns fixed integer values of energy to each colored spike in the system, i.e., instead of a colored spike $a \in U$ we consider the pair $[a,f(a)]$ with $f(a) \in \mathbf{Z}$. We extend f in the natural way to multisets over U.

The energy balance of a spiking rule $i:E/u \rightarrow P$ then is $z = f(v) - f(u)$ where v is the union of all multisets w_l in P. Such variants of extended spiking neural P systems will be called *symbol energy-controlled* ECSNPSs.

In the second variant, the energy is directly assigned to the rules only, and we write the spiking rule as $i:E/u \rightarrow P <z>$ where z is the assigned integer energy value (*rule energy-controlled* ECSNPSs).

ECSNPS with Energy Control

In the case of *energy-controlled* ESNP systems we now will only consider spiking rules i:E/u →P with E being the set of all multisets over U, i.e., we can omit E and simply write the spiking rule as i:u →P and i:u →P <z>, respectively.

In order to control the application of rules, we will impose the condition that the resulting energy balance of the applied spiking rules must be minimal.

Computational Completeness Results

Theorem 1. *The computations of any register machine can be simulated by an ECSNPS in only one node.*

Proof. Consider an arbitrary d-register machine $M = (d,B,l_0,l_h,P)$. For each register i, $1 \leq i \leq d$, of M, we use a different colored spike a_i. An additional colored spike a_0 is used to encode the label p in B={1,...,m}, i.e., p is encoded by a_0^p. We here denote $U:=\{a_i : 1 \leq i \leq d\}$.

ECSNPS with only one neuron Π = ({1},I,R):

An increment instruction p:(ADD(r),q,s) then can be simulated by the two spiking rules

$1: \{a_0^p\}U^o / a_0^p \rightarrow (1, a_r\, a_0^q)$ and

$1: \{a_0^p\}U^o / a_0^p \rightarrow (1, a_r\, a_0^s)$.

Computational Completeness Results

A zero-test and decrement instruction p:(SUB(r),q,s) then can be simulated by the two spiking rules

$1: \{a_0^p\}(U\backslash\{a_r\})^o / a_0^p \rightarrow (1, a_0^s)$ and

$1: \{a_0^p a_r\}U^o / a_0^p a_r \rightarrow (1, a_0^q)$.

Assuming that the halt instruction has label m, we finally get the simulation of the halt instruction as

$1: \{a_0^m\}U^o / a_0^m \rightarrow \lambda$.

After the application of this rule, no spike a_0 is present any more, hence, no spiking rule can be applied, i.e., the ESNP system Π halts.

Computational Completeness Results

Theorem 2. *The computations of any register machine can be simulated by an ECSNPS with symbol or rule energy control in only one node.*

Proof.

Again we consider an arbitrary d-register machine $M = (d,B,l_0,l_h,P)$. For each register i, $1 \leq i \leq d$, of M, we use a different colored spike a_i with 1 being the energy assigned to it. An additional colored spike a_0 with energy 2 is used to encode the label p in B={1,...,m}, i.e., p is encoded by a_0^p. Moreover, we use a special energy spike e with energy -1 to balance the spiking rules.

Computational Completeness Results

symbol energy-controlled ECSNPS $\Pi = (\{1\}, I, R)$:

p:(ADD(r),q,s) simulated by

$1: [a_0,2]^p \rightarrow [a_r,1]\ [a_0,2]^q\ [e,-1]^{2q+1}\ < -2p >$ and

$1: [a_0,2]^p \rightarrow [a_r,1]\ [a_0,2]^s\ [e,-1]^{2s+1}\ < -2p >$.

p:(SUB(r),q,s) simulated by

$1: [a_0,2]^p \rightarrow [a_0,2]^s\ [e,-1]^{2s}\ < -2p >$ and

$1: [a_0,2]^p\ [a_r,1] \rightarrow [a_0,2]^q\ [e,-1]^{2s}\ < -2p-1 >$.

Halt instruction simulated by the spiking rule

$1: [a_0,2]^m \rightarrow \lambda\ < -2m >$.

Elimination of the "garbage" of spikes e by

$1: [e,-1] \rightarrow \lambda\ < 1 >$.

ECSNPS with Request (ECSNPRS)

In an ECSNPS with request (ECSNPRS) spikes are requested from other neurons instead of being sent. The spiking rules then are of the form i:E/u ←P where E is the regular checking set over the set of colored spikes U, u is a finite multiset over U eliminated by this rule, and P is a (possibly empty) set of requesting productions of the form (n,w_n), $n \in \{0,1,\ldots,m\}$, w_n is a finite multiset of colored spikes over U requested along the axon from neuron n to neuron i.

To increase the number of spikes in the system, we use rules requesting spikes from the environment neuron n=0 where all spikes are assumed to be available in an unbounded number.

Computational Completeness for ECSNPRS

Theorem 3. *The computations of any register machine can be simulated by an ECSNPRS in only one actor neuron.*

Proof. Consider an arbitrary d-register machine $M = (d,B,l_0,l_h,P)$. For each register i, $1 \leq i \leq d$, of M, we use a different colored spike a_i. An additional colored spike a_0 is used to encode the label p in B={1,...,m}, i.e., p is encoded by a_0^p. We here denote $U:=\{a_i : 1 \leq i \leq d\}$.

ECSNPS with only one actor neuron Π = ({1},I,R):

An increment instruction p:(ADD(r),q,s) then can be simulated by the two spiking rules

$1: \{a_0^p\}U^o / a_0^p \leftarrow (0, a_r\ a_0^q)$ and

$1: \{a_0^p\}U^o / a_0^p \leftarrow (0, a_r\ a_0^s)$.

Computational Completeness for ECSNPRS

A zero-test and decrement instruction p:(SUB(r),q,s) then can be simulated by the two spiking rules

$1: \{a_0^p\}(U\backslash\{a_r\})^o / a_0^p \leftarrow (0,a_0^s)$ and

$1: \{a_0^p a_r\}U^o / a_0^p a_r \leftarrow (0,a_0^q)$.

Assuming that the halt instruction has label m, we finally get the simulation of the halt instruction as

$1: \{a_0^m\}U^o / a_0^m \leftarrow \lambda$.

After the application of this rule, no spike a_0 is present any more, hence, no spiking rule can be applied, i.e., the ESNP system Π halts.

The Rudi-Gheorghe Challenge of 2017:

Prove that one coloured spike is not enough with our definition for an ECSNPRS to obtain computational completeness.

REWARD:

(65 – age) Euros

Co-authorship

Condition to be fulfilled for obtaining the Reward:

Proof approved by Rudi and Gheorghe!