A small universal spiking neural P system

Turlough Neary

Boole Centre for Research in Informatics,
University College Cork, Ireland,

Funded by Science Foundation Ireland Research
Frontiers Programme grant number 07/RFP/CSMF641.

August 27, 2008

Introduction

- In this talk we present two small universal spiking neural P systems.
- Spiking neural P systems are the result of a synergy inspired by spiking neural networks and P systems.
- These systems were first presented and proved universal in 2006 by Ionescu, Păun and Yokomori.

Previous small spiking neural P systems

- Păun and Păun gave a strongly universal spiking neural P system with 84 neurons and another that has extended rules with 49 neurons.
- Subsequently, the number of neurons used for strong universality was reduced from 84 to 67 and from 49 to 41 by Zhang et al.
- Recently we proved that there exists no universal (extended) spiking neural P system that simulates Turing machines in less then exponential time and space.
- A universal spiking neural P system with exhaustive use of extended rules has been given that simulates Turing machines in polynomial time. This system has only 18 neurons.

Our results

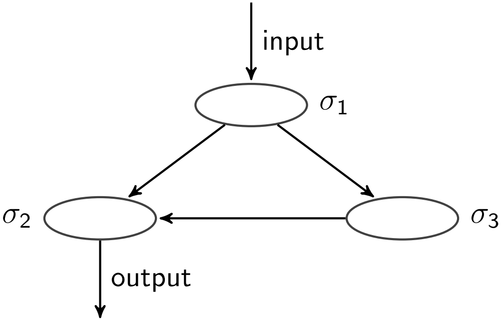
- Here we present a weakly universal spiking neural P system that has extended rules and only 12 neurons. This system simulates a weakly universal 2 register machine.
- By adapting our algorithm we can simulated more general register machines with more registers. We show that there exists a strongly universal spiking neural P system that has extended rules and 18 neurons.

Extended spiking neural P systems

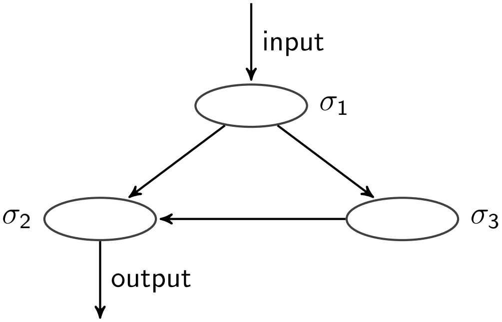
A spiking extended neural P system is a tuple
$\Pi = (O, \sigma_1, \sigma_2, \cdots, \sigma_m, syn, in, out)$, where:

1. $O = \{s\}$ is the unary alphabet (s is known as a spike),
2. $\sigma_1, \sigma_2, \cdots, \sigma_m$ are neurons, of the form $\sigma_i = (n_i, R_i), 1 \leqslant i \leqslant m$, where:
   - 2.1 $n_i \geqslant 0$ is the initial number of spikes contained in σ_i,
   - 2.2 R_i is a finite set of rules of the following two forms:
     - 2.2.1 $E/s^b \rightarrow s^c; d$, where E is a regular expression over s, $b \geqslant c \geqslant 1$ and $d \geqslant 0$,
     - 2.2.2 $s^e \rightarrow \lambda$, where λ is the empty word, $e \geqslant 1$, and for all $E/s^b \rightarrow s; d$ from R_i $s^e \notin L(E)$ where $L(E)$ is the language defined by E,
3. $syn \subseteq \{1, 2, \cdots, m\} \times \{1, 2, \cdots, m\}$ is the set of synapses between neurons, where $i \neq j$ for all $(i, j) \in syn$,
4. $in, out \in \{\sigma_1, \sigma_2, \cdots, \sigma_m\}$ are the input and output neurons, respectively.

Extended spiking neural P system



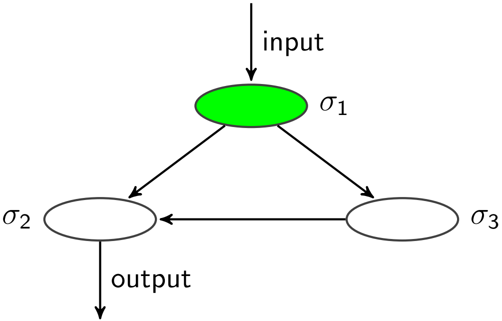
An extended spiking neural P systems executing rule $E/s^b \longrightarrow s^c; d$



$t_1 : \sigma_1 = 4,$ $\quad$ $(s^2)^*/s^3 \longrightarrow s^2; 0.$

On the left $\sigma_k = y$ gives the number y of spikes in neuron σ_k at time t_j and on the right is the next rule that is to be applied at time t_1 if there is an applicable rule at that time.

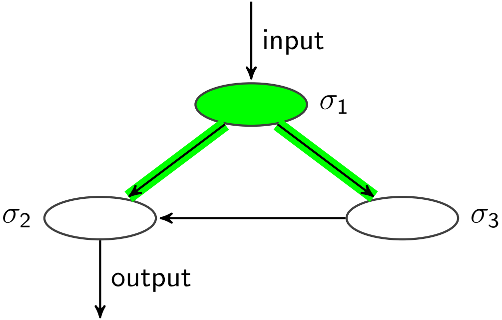
An extended spiking neural P systems executing rule $E/s^b \rightarrow s^c; d$



$t_1 : \sigma_1 = 4, \qquad (s^2)^*/s^3 \rightarrow s^2; 0.$

On the left $\sigma_k = y$ gives the number y of spikes in neuron σ_k at time t_j and on the right is the next rule that is to be applied at time t_1 if there is an applicable rule at that time.

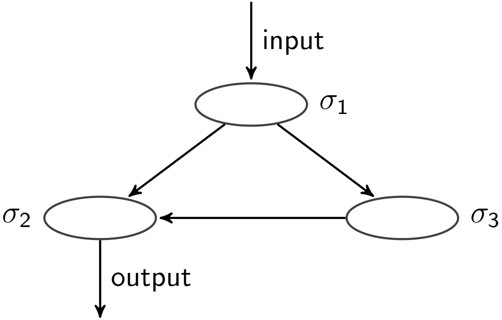
An extended spiking neural P systems executing rule $E/s^b \longrightarrow s^c; d$



$t_1 : \sigma_1 = 4,$ $\qquad (s^2)^*/s^3 \longrightarrow s^2; 0.$

On the left $\sigma_k = y$ gives the number y of spikes in neuron σ_k at time t_j and on the right is the next rule that is to be applied at time t_1 if there is an applicable rule at that time.

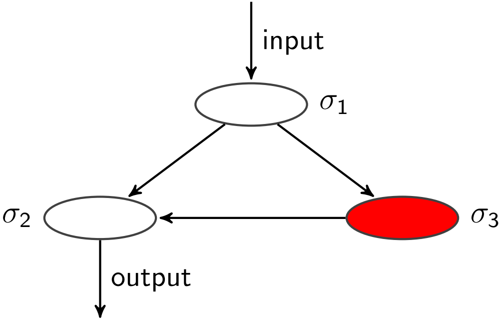
An extended spiking neural P systems executing rule $E/s^b \to s^c; d$



$$t_2 : \sigma_1 = 1, \quad \sigma_2 = 2, \quad \sigma_3 = 2,$$

$$s^2 \to \lambda$$

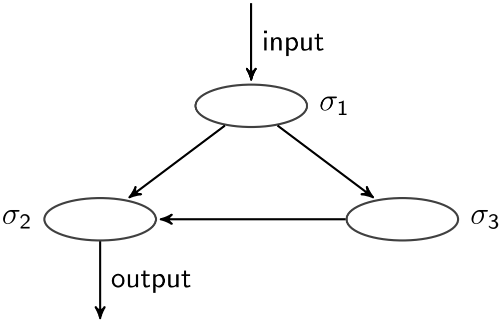
An extended spiking neural P systems executing rule $s^e \rightarrow \lambda$



$$t_2 : \sigma_1 = 1, \quad \sigma_2 = 2, \quad \sigma_3 = 2,$$

$$s^2 \rightarrow \lambda$$

An extended spiking neural P systems executing rule $s^e \to \lambda$



$$
\begin{aligned}
t_3 : \sigma_1 &= 1, \\
\sigma_2 &= 2, \\
\sigma_3 &= 0,
\end{aligned}
$$

Register machine definition

A register machine is a tuple $C = (z, r_1, r_m, Q, q_1, q_h)$, where z gives the number of registers, r_1 and r_m are the input and output registers respectively, $Q = \{q_1, q_2, \cdots, q_h\}$ is the set of instructions, $q_1, q_h \in Q$ are the initial and halt instructions, respectively.

Register machine operation

Each register r_j stores a natural number value $x \geqslant 0$. Each instruction q_i is of one of the following two forms $q_i : INC(j)$ or $q_i : DEC(j)q_k$, and is executed as follows:

- $q_i : INC(j)$ increment the value x stored in register r_j by 1 and move to instruction q_{i+1}.
- $q_i : DEC(j)q_k$ if the value x stored in register r_j is greater than 0 then decrement this value by 1 and move to instruction q_{i+1}, otherwise if $x = 0$ move to instruction q_k.

Korec's notions of universality for register machines

Let $(\phi_0, \phi_1, \phi_2, \ldots)$ be a Gödel enumeration of all unary partial recursive functions. A register machine U is weakly universal if $\phi_x(y) = f(U(g(x, y)))$ or $\phi_x(y) = U(g(x, y))$ where g and f are recursive functions. A register machine U is strongly universal if $\phi_x(y) = U(x, y)$ where g and f are recursive functions.

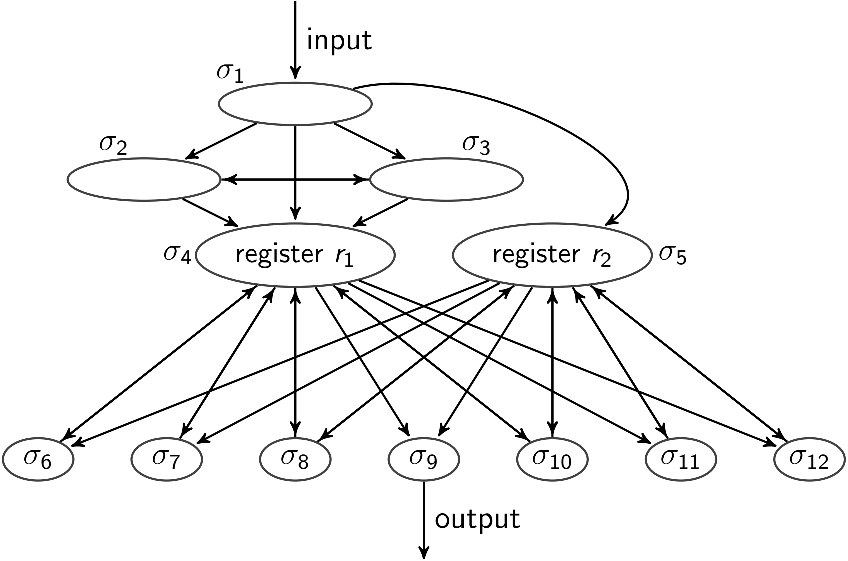
Universality of 2 register machines

In his book “Finite and infinite machines”, Minsky proves the universality 2 register machines by showing that they simulate Turing machines. If we use Minsky’s algorithm to construct a universal 2 register machine it will in fact be weakly universal.

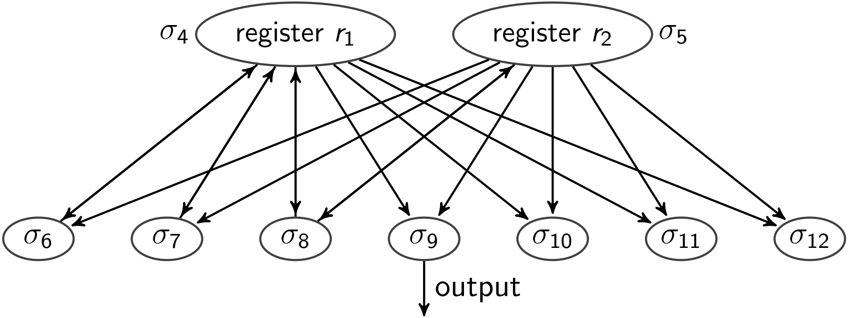
A weakly universal spiking neural P system

We give a universal spiking neural P system that simulates a universal 2 register machine. Using Minsky's algorithm to encode a Turing machine, and its input, as input to our spiking neural P system we get recursive encoding and decoding functions.

Small weakly universal spiking neural P system

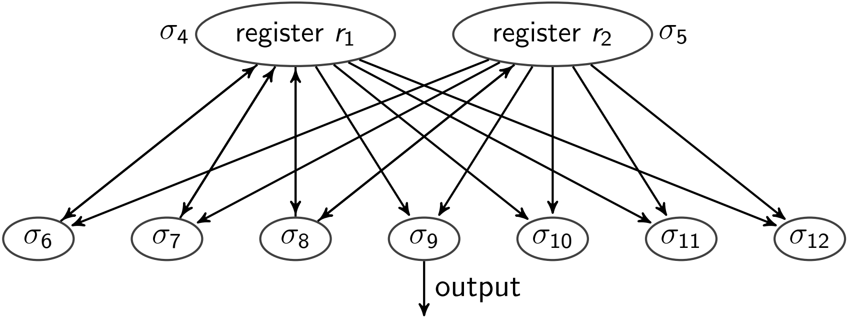


Encoding



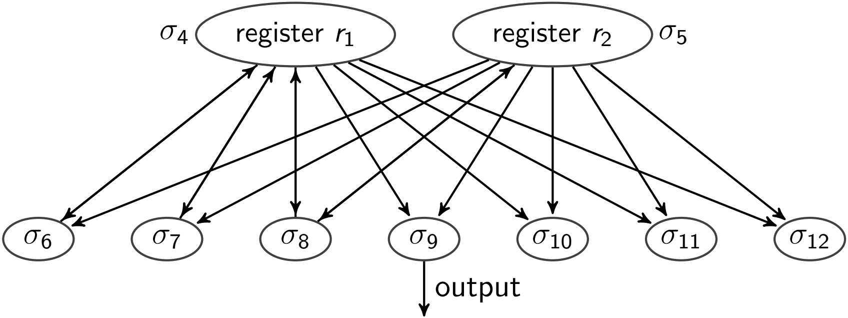
σ_4
register r_1
register r_2
σ_5
σ_6
σ_7
σ_8
σ_9
σ_{10}
σ_{11}
σ_{12}
output

Encoding



Let x_1 and x_2 be the values stored in the registers r_1 and r_2, respectively. Then x_1 and x_2 are stored as $4hx_1$ and $4hx_2$ spikes in neurons σ_4 and σ_5, respectively. The next instruction q_i to be executed is stored in each of the neurons σ_4 and σ_5 as $2(h+i)$ spikes.

Simulate $q_i : INC(1)$

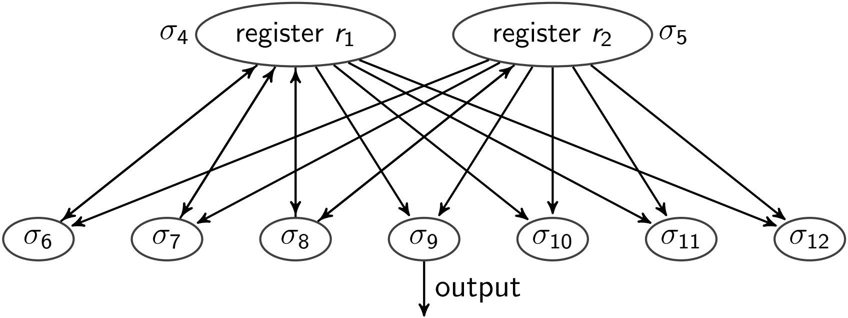


$$t_j : \sigma_4 = 4hx_1 + 2(h+i), \qquad s^{2(h+i)}(s^{4h})^*/s^{2(h+i)} \rightarrow s^{2(h+i)-1}; 0,$$

$$\sigma_5 = 4hx_2 + 2(h+i), \qquad s^{2(h+i)}(s^{4h})^*/s^{2(h+i)} \rightarrow s; 0.$$

On the left $\sigma_k = y$ gives the number y of spikes in neuron σ_k at time t_j and on the right is the next rule that is to be applied at time t_j if there is an applicable rule at that time.

Simulate $q_i : INC(1)$

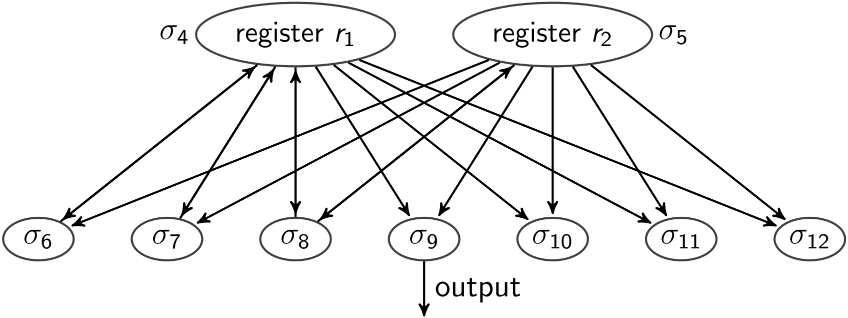


$$t_j : \sigma_4 = 4hx_1 + 2(h+i), \qquad s^{2(h+i)}(s^{4h})^*/s^{2(h+i)} \rightarrow s^{2(h+i)-1}; 0,$$

$$\sigma_5 = 4hx_2 + 2(h+i), \qquad s^{2(h+i)}(s^{4h})^*/s^{2(h+i)} \rightarrow s; 0.$$

On the left $\sigma_k = y$ gives the number y of spikes in neuron σ_k at time t_j and on the right is the next rule that is to be applied at time t_j if there is an applicable rule at that time.

Simulate $q_i : INC(1)$

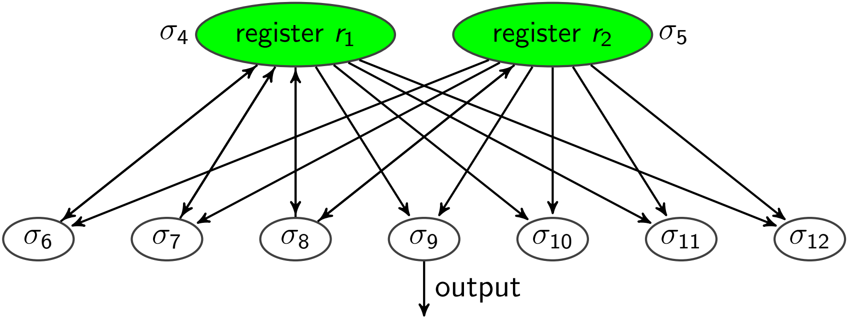


$$t_j : \sigma_4 = 4hx_1, \qquad s^{2(h+i)}(s^{4h})^*/s^{2(h+i)} \rightarrow s^{2(h+i)-1}; 0,$$

$$\sigma_5 = 4hx_2, \qquad s^{2(h+i)}(s^{4h})^*/s^{2(h+i)} \rightarrow s; 0.$$

On the left $\sigma_k = y$ gives the number y of spikes in neuron σ_k at time t_j and on the right is the next rule that is to be applied at time t_j if there is an applicable rule at that time.

Simulate $q_i : INC(1)$



$t_j : \sigma_4 = 4hx_1,$

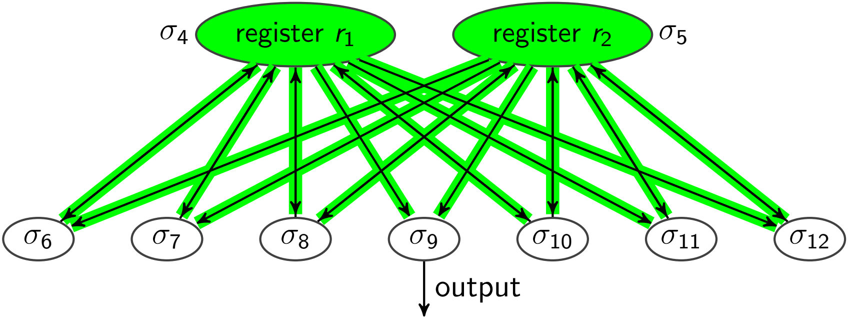
$\sigma_5 = 4hx_2,$

$$s^{2(h+i)}(s^{4h})^*/s^{2(h+i)} \rightarrow s^{2(h+i)-1}; 0,$$

$$s^{2(h+i)}(s^{4h})^*/s^{2(h+i)} \rightarrow s; 0.$$

On the left $\sigma_k = y$ gives the number y of spikes in neuron σ_k at time t_j and on the right is the next rule that is to be applied at time t_j if there is an applicable rule at that time.

Simulate q_i : $INC(1)$

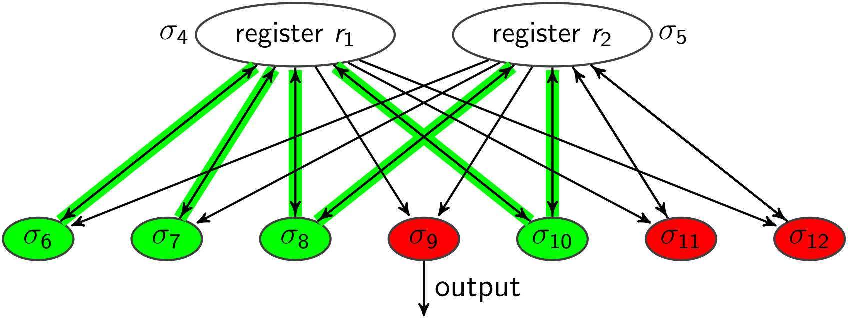


$t_j : \sigma_4 = 4hx_1,$ $\quad s^{2(h+i)}(s^{4h})^*/s^{2(h+i)} \to s^{2(h+i)-1}; 0,$

$\sigma_5 = 4hx_2,$ $\quad s^{2(h+i)}(s^{4h})^*/s^{2(h+i)} \to s; 0.$

On the left $\sigma_k = y$ gives the number y of spikes in neuron σ_k at time t_j and on the right is the next rule that is to be applied at time t_j if there is an applicable rule at that time.

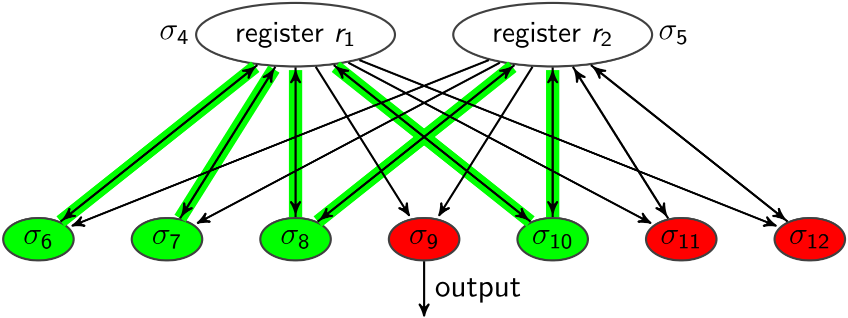
Simulate $q_i : INC(1)$



$$
\begin{aligned}
t_{j+1} : \sigma_4 &= 4hx_1, \\
\sigma_5 &= 4hx_2, \\
\sigma_6, \sigma_7, \sigma_8 &= 2(h+i), \\
\sigma_9, \sigma_{11}, \sigma_{12} &= 2(h+i), \\
\sigma_{10} &= 2(h+i),
\end{aligned}
$$

$$
\begin{aligned}
s^{2(h+i)}/s^{2(h+i)} &\to s^{2h}; 0, \\
s^{2(h+i)} &\to \lambda, \\
s^{2(h+i)}/s^{2(h+i)} &\to s^{2(i+1)}; 0.
\end{aligned}
$$

Simulate $q_i : INC(1)$



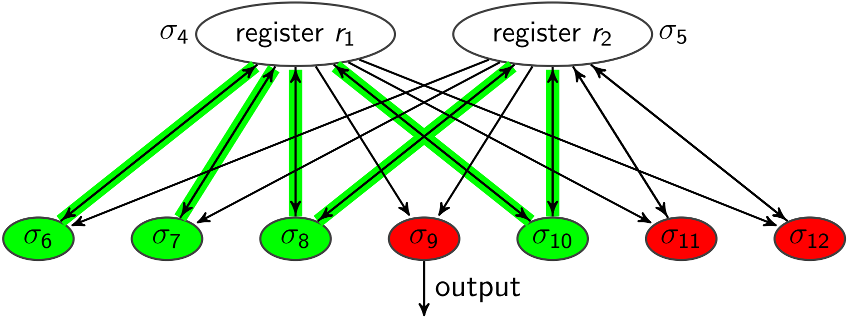
$$t_{j+1} : \sigma_4 = 4hx_1,$$

$$\sigma_5 = 4hx_2,$$

$$\sigma_6, \sigma_7, \sigma_8 = 2(h+i), \qquad s^{2(h+i)}/s^{2(h+i)} \rightarrow s^{2h}; 0,$$

$$\sigma_{10} = 2(h+i), \qquad s^{2(h+i)}/s^{2(h+i)} \rightarrow s^{2(i+1)}; 0.$$

Simulate q_i : $INC(1)$



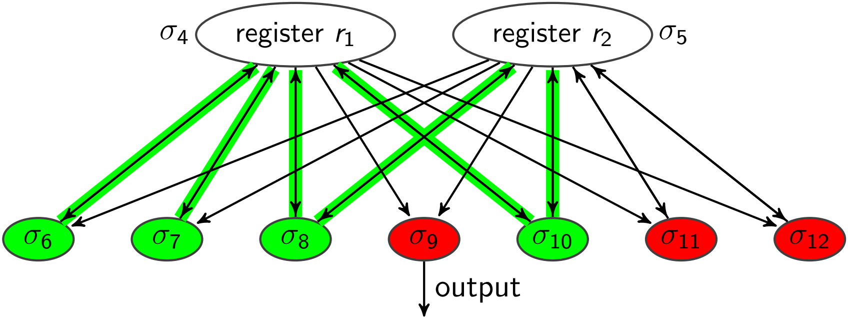
$t_{j+1} : \sigma_4 = 4hx_1,$

$\sigma_5 = 4hx_2,$

$\sigma_6, \sigma_7, \sigma_8 = 2(h+i),$ $\qquad s^{2(h+i)}/s^{2(h+i)} \rightarrow s^{2h}; 0,$

$\sigma_{10} = 2(h+i),$ $\qquad s^{2(h+i)}/s^{2(h+i)} \rightarrow s^{2(i+1)}; 0.$

Simulate $q_i : INC(1)$



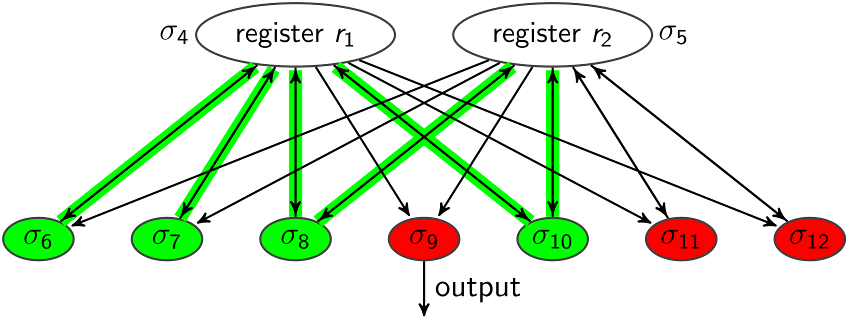
$$t_{j+1} : \sigma_4 = 4h(x_1+1),$$

$$\sigma_5 = 4hx_2,$$

$$\sigma_8 = 2(h+i), \qquad s^{2(h+i)}/s^{2(h+i)} \to s^{2h}; 0,$$

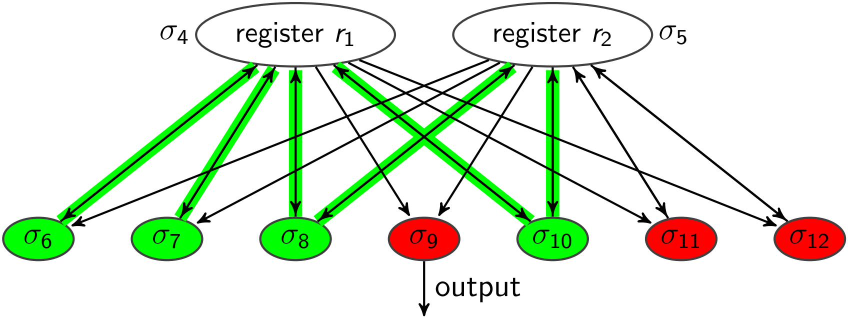
$$\sigma_{10} = 2(h+i), \qquad s^{2(h+i)}/s^{2(h+i)} \to s^{2(i+1)}; 0.$$

Simulate q_i : $INC(1)$



$$t_{j+1} : \sigma_4 = 4h(x_1 + 1),$$
$$\sigma_5 = 4hx_2,$$
$$\sigma_8 = 2(h + i), \qquad s^{2(h+i)}/s^{2(h+i)} \rightarrow s^{2h}; 0,$$
$$\sigma_{10} = 2(h + i), \qquad s^{2(h+i)}/s^{2(h+i)} \rightarrow s^{2(i+1)}; 0.$$

Simulate $q_i : INC(1)$



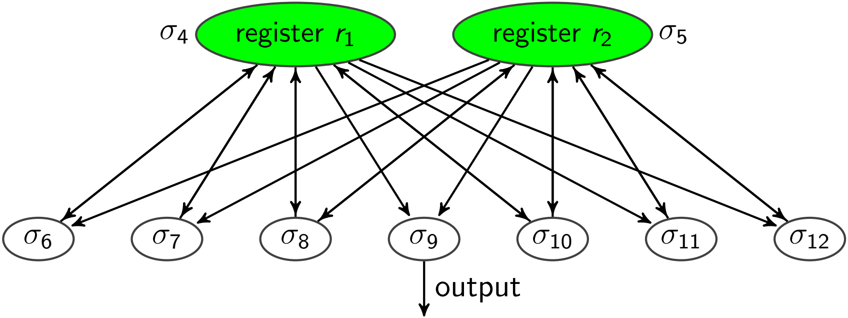
$t_{j+1} : \sigma_4 = 4h(x_1 + 1),$

$\sigma_5 = 4hx_2,$

$\sigma_8 = 2(h + i),$ $\quad s^{2(h+i)}/s^{2(h+i)} \to s^{2h}; 0,$

$\sigma_{10} = 2(h + i),$ $\quad s^{2(h+i)}/s^{2(h+i)} \to s^{2(i+1)}; 0.$

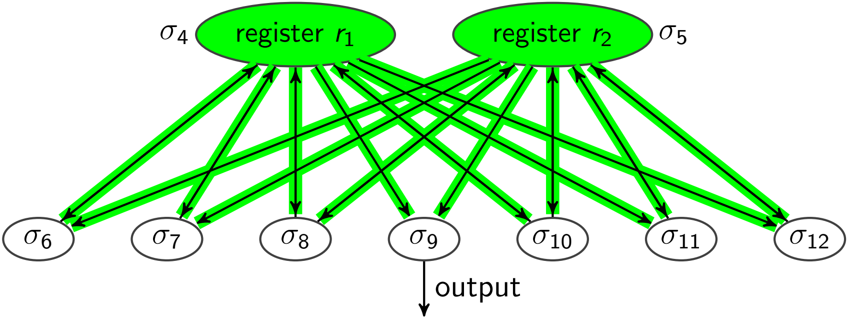
Simulate $q_i : INC(1)$



$$t_{j+2} : \sigma_4 = 4h(x_1 + 1) + 2(h + i + 1),$$
$$\sigma_5 = 4hx_2 + 2(h + i + 1).$$

At time t_{j+2} the simulation of $q_i : INC(1)$ is complete. The encoded register value has been incremented by increasing it from $4hx_1$ to $4h(x_1 + 1)$. The encoding $2(h + i + 1)$ of the next instruction q_{i+1} has been established.

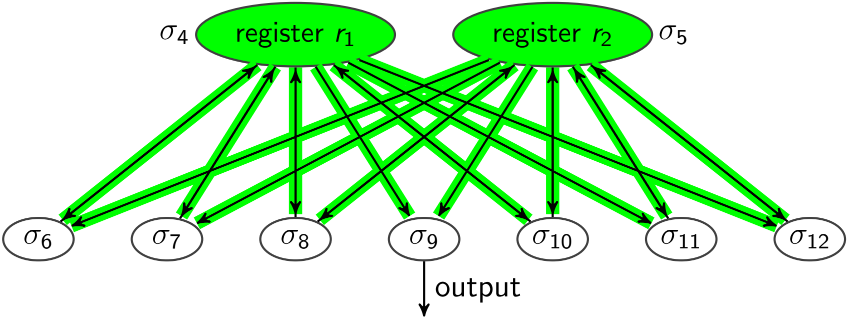
Simulate $q_i : DEC(1)q_k$ for $x_1 > 0$



$$t_j : \sigma_4 = 4hx_1 + 2(h+i), \quad s^{4h+2(h+i)}(s^{4h})^*/s^{4h+2(h+i)} \to s^{2(h+i)}; 0,$$

$$\sigma_5 = 4hx_2 + 2(h+i), \quad s^{2(h+i)}(s^{4h})^*/s^{2(h+i)} \to s; 0.$$

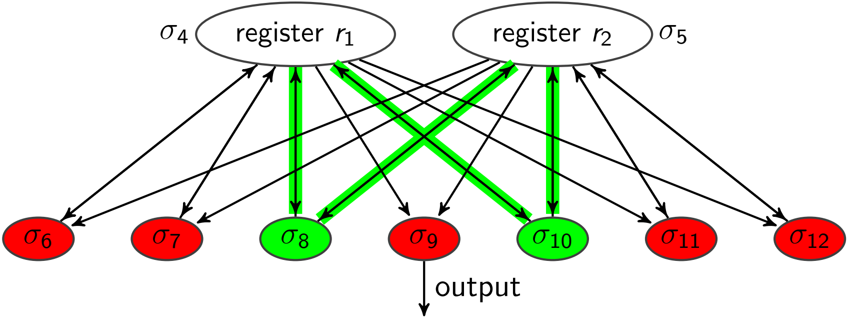
Simulate $q_i : DEC(1)q_k$ for $x_1 > 0$



$$t_j : \sigma_4 = 4hx_1 + 2(h+i), \quad s^{4h+2(h+i)}(s^{4h})^*/s^{4h+2(h+i)} \rightarrow s^{2(h+i)}; 0,$$

$$\sigma_5 = 4hx_2 + 2(h+i), \quad s^{2(h+i)}(s^{4h})^*/s^{2(h+i)} \rightarrow s; 0.$$

Simulate $q_i : DEC(1)q_k$ for $x_1 > 0$



$t_{j+1} : \sigma_4 = 4h(x_1 - 1),$

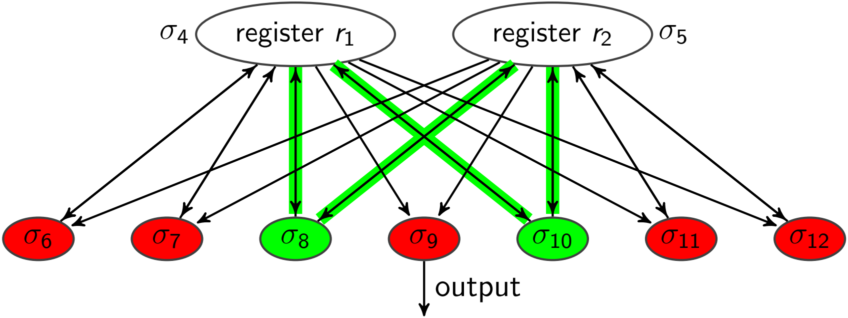
$\sigma_5 = 4hx_2,$

$\sigma_6, \sigma_7, \sigma_9, \sigma_{11}, \sigma_{12} = 2(h+i)+1,$ $\quad s^{2(h+i)+1} \to \lambda,$

$\sigma_8 = 2(h+i)+1,$ $\quad s^{2(h+i)+1}/s^{2(h+i)+1} \to s^{2h}; 0,$

$\sigma_{10} = 2(h+i)+1,$ $\quad s^{2(h+i)+1}/s^{2(h+i)+1} \to s^{2(i+1)}; 0.$

Simulate $q_i : DEC(1)q_k$ for $x_1 > 0$



$t_{j+1} : \sigma_4 = 4h(x_1 - 1),$

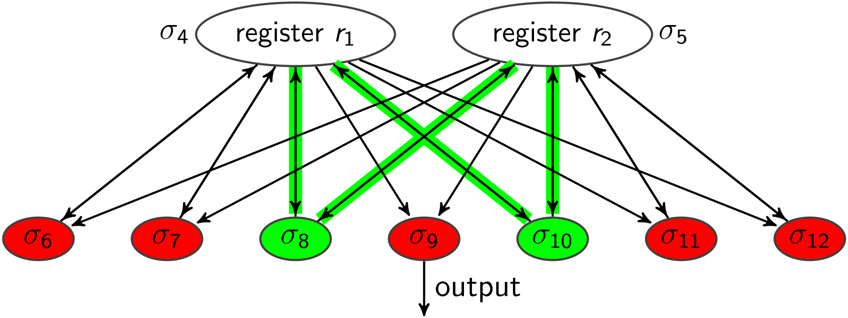
$\sigma_5 = 4hx_2,$

$\sigma_6, \sigma_7, \sigma_9, \sigma_{11}, \sigma_{12} = 2(h+i)+1,$ $\quad s^{2(h+i)+1} \to \lambda,$

$\sigma_8 = 2(h+i)+1,$ $\quad s^{2(h+i)+1}/s^{2(h+i)+1} \to s^{2h}; 0,$

$\sigma_{10} = 2(h+i)+1,$ $\quad s^{2(h+i)+1}/s^{2(h+i)+1} \to s^{2(i+1)}; 0.$

Simulate $q_i : DEC(1)q_k$ for $x_1 > 0$



$t_{j+1} : \sigma_4 = 4h(x_1 - 1),$

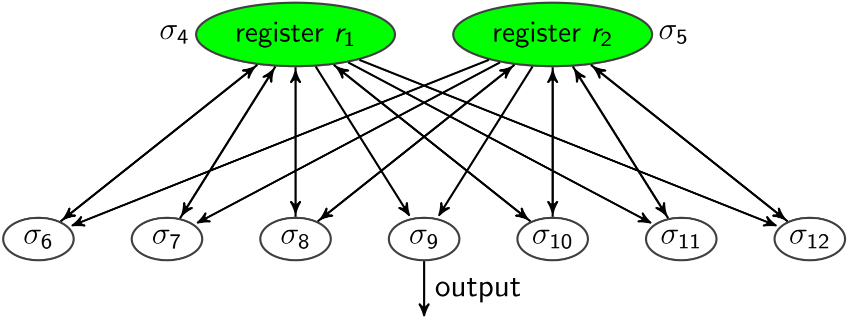
$\sigma_5 = 4hx_2,$

$\sigma_6, \sigma_7, \sigma_9, \sigma_{11}, \sigma_{12} = 2(h+i)+1, \qquad s^{2(h+i)+1} \to \lambda,$

$\sigma_8 = 2(h+i)+1, \qquad s^{2(h+i)+1}/s^{2(h+i)+1} \to s^{2h}; 0,$

$\sigma_{10} = 2(h+i)+1, \qquad s^{2(h+i)+1}/s^{2(h+i)+1} \to s^{2(i+1)}; 0.$

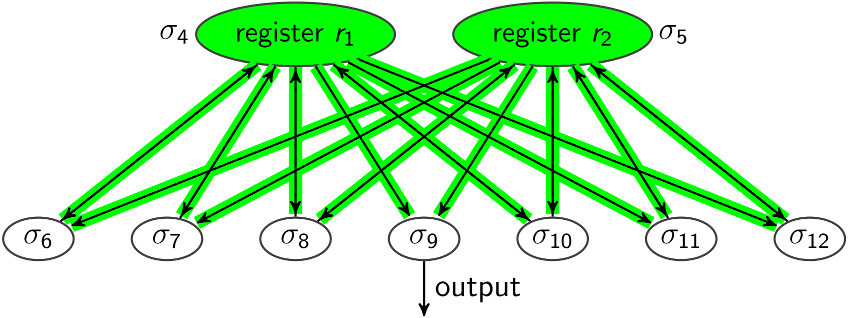
Simulate $q_i : DEC(1)q_k$ for $x_1 > 0$



$$t_{j+2} : \sigma_4 = 4h(x_1 - 1) + 2(h + i + 1),$$
$$\sigma_5 = 4hx_2 + 2(h + i + 1).$$

At time t_{j+2} the simulation of $q_i : DEC(1)$ is complete. The encoded register value has been decremented by decreasing it from $4hx_1$ to $4h(x_1 - 1)$. The encoding $2(h + i + 1)$ of the next instruction q_{i+1} has been established.

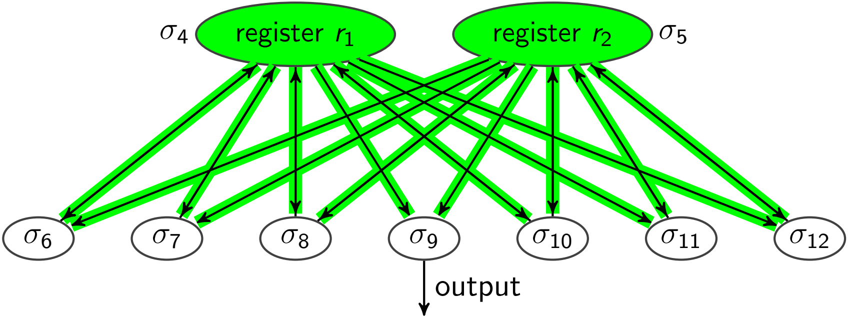
Simulate $q_i : DEC(1)q_k$ for $x_1 = 0$



$t_j : \sigma_4 = 2(h+i),$ $\quad s^{2(h+i)}/s^{2(h+i)} \to s^{2(h+i)-1}; 0,$

$\sigma_5 = 4hx_2 + 2(h+i),$ $\quad s^{2(h+i)}(s^{4h})^*/s^{2(h+i)} \to s; 0.$

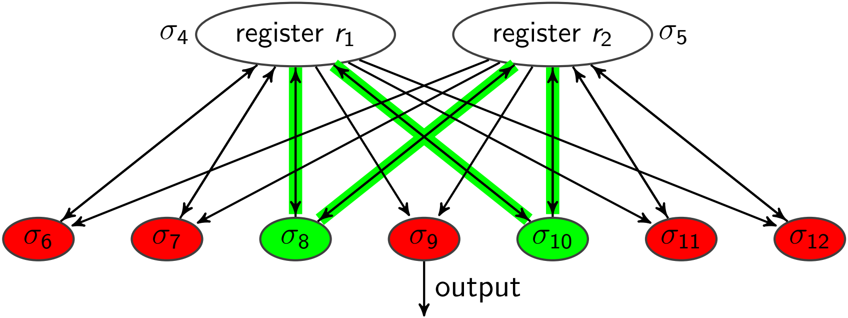
Simulate $q_i : DEC(1)q_k$ for $x_1 = 0$



$t_j : \sigma_4 = 2(h+i),$ $\quad s^{2(h+i)}/s^{2(h+i)} \to s^{2(h+i)-1}; 0,$

$\sigma_5 = 4hx_2 + 2(h+i),$ $\quad s^{2(h+i)}(s^{4h})^*/s^{2(h+i)} \to s; 0.$

Simulate $q_i : DEC(1)q_k$ for $x_1 = 0$



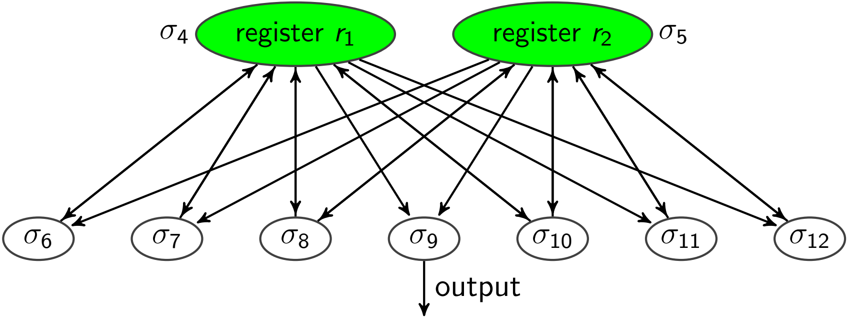
$t_{j+1} : \sigma_5 = 4hx_2,$

$\sigma_6, \sigma_7, \sigma_9, \sigma_{11}, \sigma_{12} = 2(h+i),$ $\qquad s^{2(h+i)} \rightarrow \lambda,$

$\sigma_8 = 2(h+i),$ $\qquad s^{2(h+i)}/s^{2(h+i)} \rightarrow s^{2h}; 0,$

$\sigma_{10} = 2(h+i),$ $\qquad s^{2(h+i)}/s^{2(h+i)} \rightarrow s^{2k}; 0.$

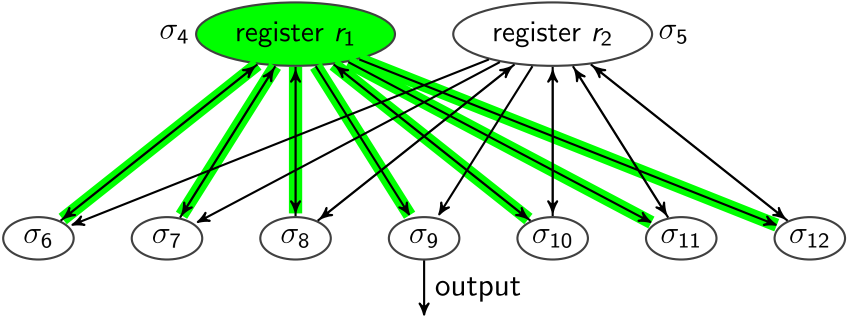
Simulate $q_i : DEC(1)q_k$ for $x_1 = 0$



$$t_{j+2} : \sigma_4 = 2(h+k),$$
$$\sigma_5 = 4hx_2 + 2(h+k).$$

Note that at time t_{j+2}, when the simulation is complete, the encoding $2(h+k)$ of the next instruction q_{i+1} has been established.

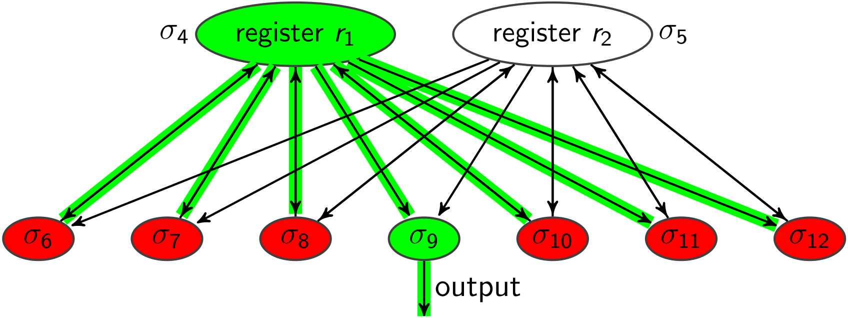
Halting and output



$$t_j : \sigma_4 = 4hx_1 + 2h + 3, \qquad s^{6h+3}(s^{4h})^*/s^{6h} \to s; 0,$$

$$\sigma_5 = 4hx_2 + 2h + 3.$$

Halting and output



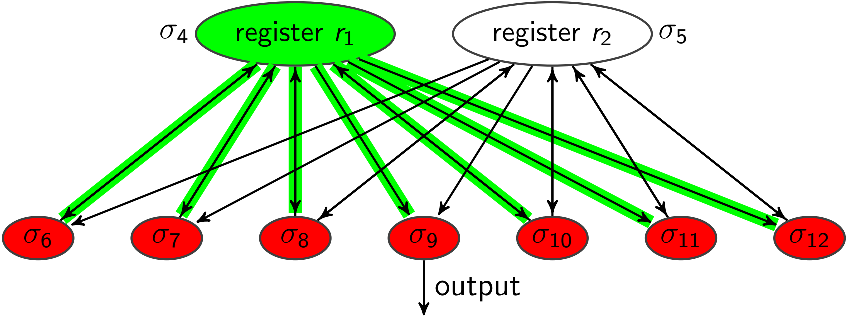
$t_{j+1}: \sigma_4 = 4h(x_1 - 1) + 3, \qquad s^7(s^{4h})^*/s^{4h} \to s^{2h}; 0,$

$\sigma_5 = 4hx_2 + 2h + 3,$

$\sigma_6, \sigma_7, \sigma_8, \sigma_{10}, \sigma_{11}, \sigma_{12} = 1, \qquad s \to \lambda,$

$\sigma_9 = 1, \qquad s/s \to s; 0.$

Halting and output

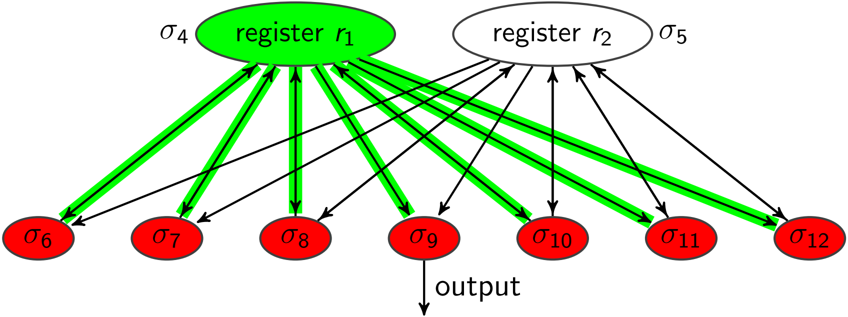


$$t_{j+2}: \sigma_4 = 4h(x_1 - 2) + 3, \qquad s^7(s^{4h})^*/s^{4h} \to s^{2h}; 0,$$

$$\sigma_5 = 4hx_2 + 2h + 3,$$

$$\sigma_6, \sigma_7, \sigma_8, \sigma_9, \sigma_{10}, \sigma_{11}, \sigma_{12} = 2h, \qquad s^{2h} \to \lambda.$$

Halting and output

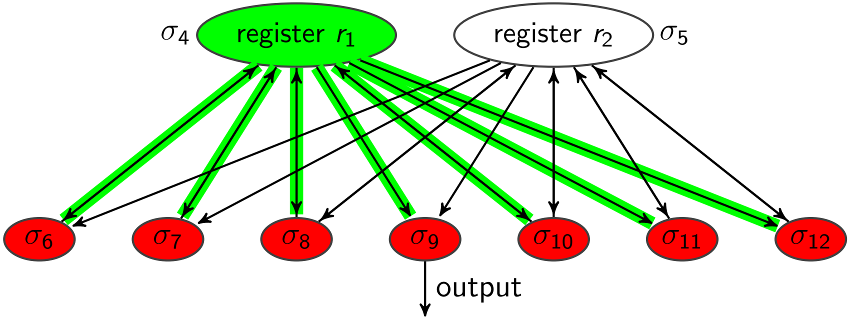


$$t_{j+2} : \sigma_4 = 4h(x_1 - 2) + 3, \qquad s^7(s^{4h})^*/s^{4h} \to s^{2h}; 0,$$

$$\sigma_5 = 4hx_2 + 2h + 3,$$

$$\sigma_6, \sigma_7, \sigma_8, \sigma_9, \sigma_{10}, \sigma_{11}, \sigma_{12} = 2h, \qquad s^{2h} \to \lambda.$$

Halting and output

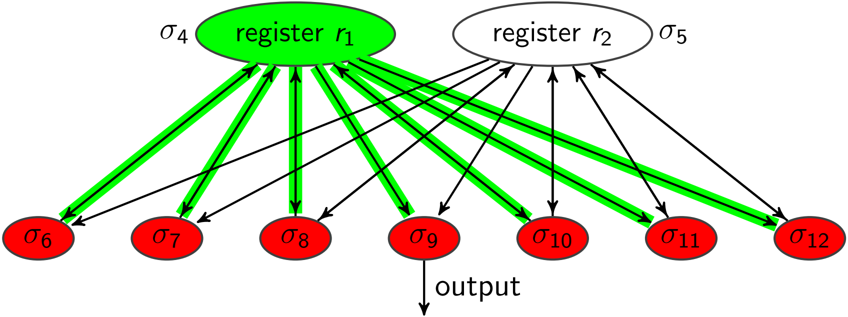


$$t_{j+2} : \sigma_4 = 4h(x_1 - 2) + 3, \qquad s^7(s^{4h})^* / s^{4h} \to s^{2h}; 0,$$

$$\sigma_5 = 4hx_2 + 2h + 3,$$

$$\sigma_6, \sigma_7, \sigma_8, \sigma_9, \sigma_{10}, \sigma_{11}, \sigma_{12} = 2h, \qquad s^{2h} \to \lambda.$$

Halting and output

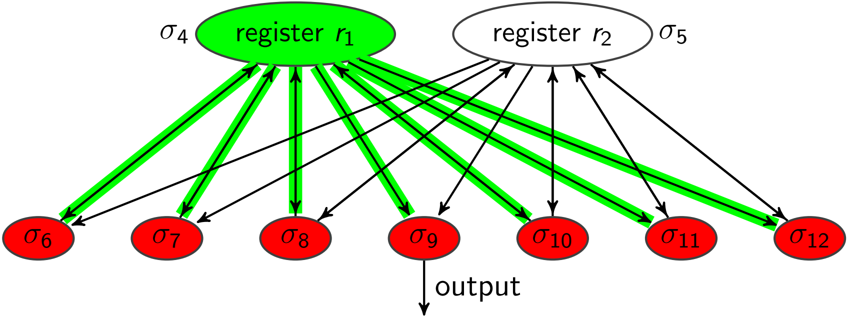


$t_{j+x_1}: \sigma_4 = 3,$ $\qquad s^3/s^3 \to s; 0,$

$\sigma_5 = 4hx_2 + 2h + 3,$

$\sigma_6, \sigma_7, \sigma_8, \sigma_9, \sigma_{10}, \sigma_{11}, \sigma_{12} = 2h,$ $\qquad s^{2h} \to \lambda.$

Halting and output

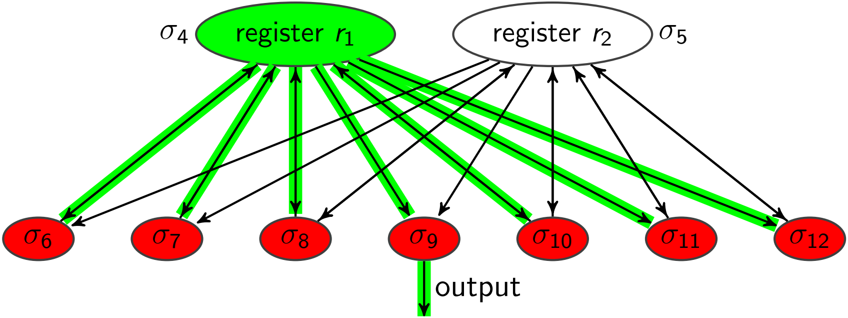


t_{j+x_1} : $\sigma_4 = 3$, $\quad s^3/s^3 \rightarrow s; 0$,

$\sigma_5 = 4hx_2 + 2h + 3$,

$\sigma_6, \sigma_7, \sigma_8, \sigma_9, \sigma_{10}, \sigma_{11}, \sigma_{12} = 2h$, $\quad s^{2h} \rightarrow \lambda$.

Halting and output

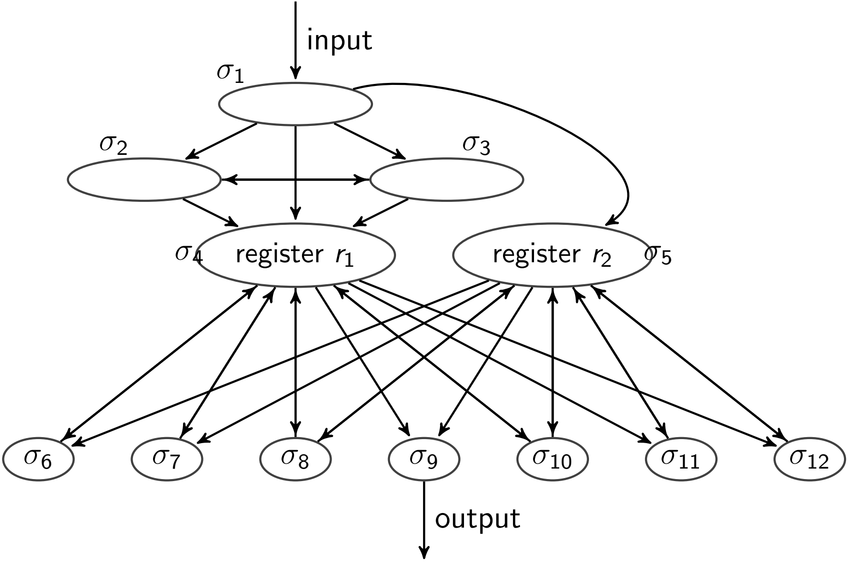


$t_{j+x_1+1}: \sigma_5 = 4hx_2 + 2h + 3,$

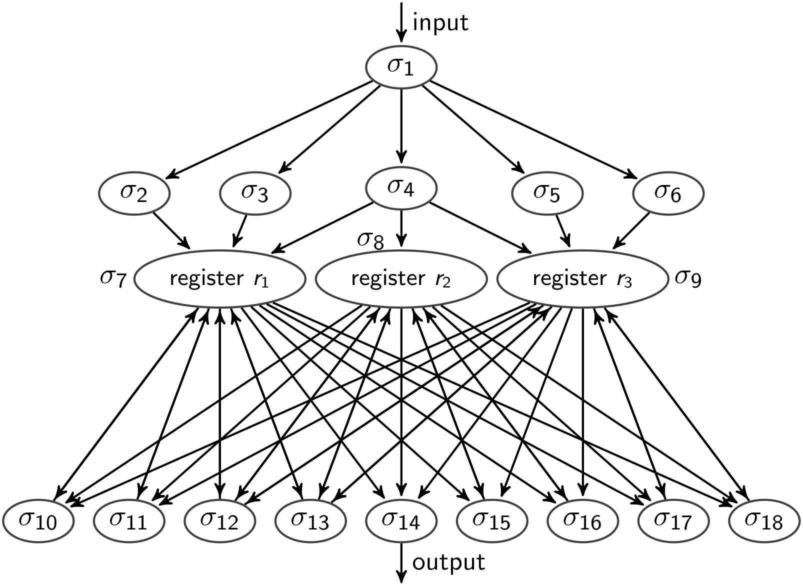
$\sigma_6, \sigma_7, \sigma_8, \sigma_{10}, \sigma_{11}, \sigma_{12} = 1,$ $\quad s \to \lambda,$

$\sigma_9 = 1,$ $\quad s/s \to s; 0.$

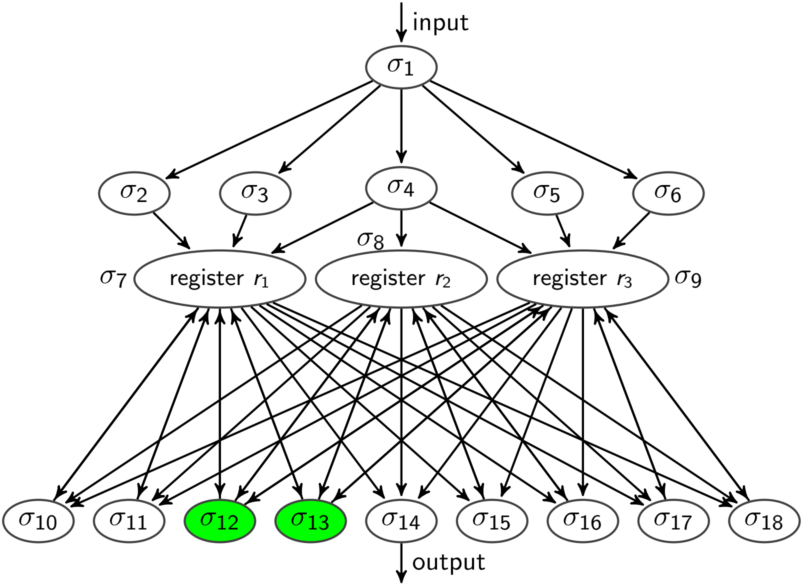
Small weakly universal spiking neural P system



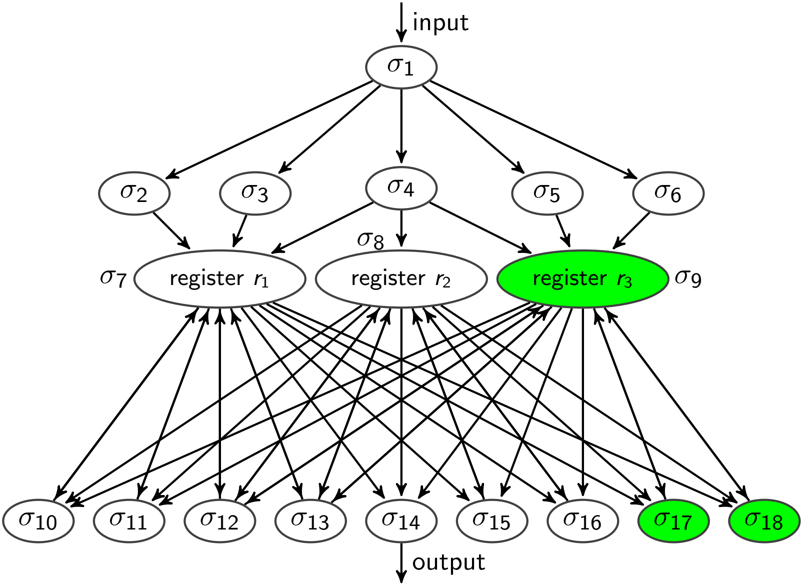
Small strongly universal spiking neural P system



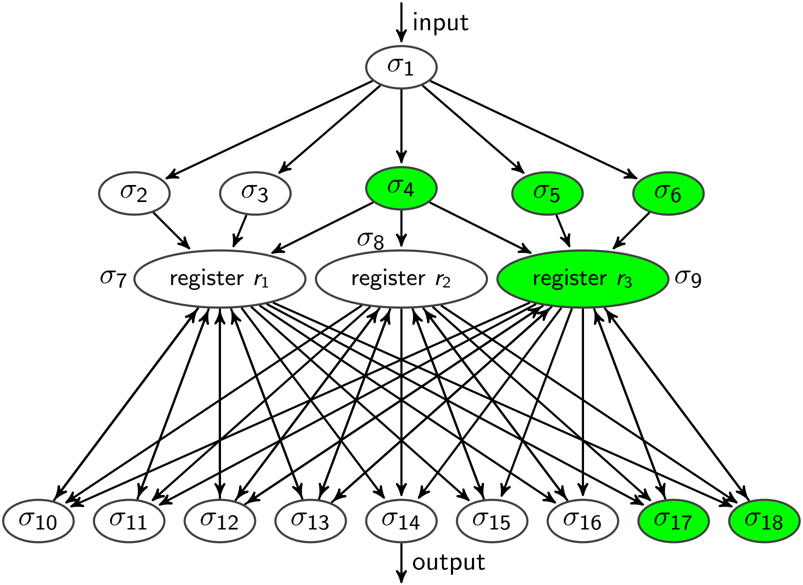
Small strongly universal spiking neural P system



Small strongly universal spiking neural P system



Small strongly universal spiking neural P system



Conclusions

- We have given an extended spiking neural P system with 12 neurons that is weakly universal and another with 18 neurons that is strongly universal.
- We have given a new simulation technique of register machines for spiking neural P system.
- The simulation technique given for our spiking neural P systems is easily adapted to simulate more general register machines.
- There exist spiking neural P systems with 8 neurons which have undecidable reachability questions.