On Some Recent Efforts In Spiking Computations[1]

Francis George Carreon-Cabarle

Algorithms & Complexity Lab,
Dept. of Computer Science,
University of the Philippines Diliman
fccabarle@up.edu.ph

[1]PhD scholarship by the ERDT program of the Department of Science and Technology, Philippines.

Seminar Main Parts

- Part 1: (Very) Short informal introduction
- Part 2: Some previous works
- Part 3: SNP systems and structural plasticity
- Part 4: Further ideas

Part 1: (Very) Short informal introduction

Algorithms & Complexity lab (aclab.dcs.upd.edu.ph)

6th BIC-TA (Malaysia, 2011)

CMC13 (Hungary, 2012)

Part 2: Some previous works

SNP systems simulations in GPUs (BWMC 2011, CMC 2011, ROMJIST 2012)

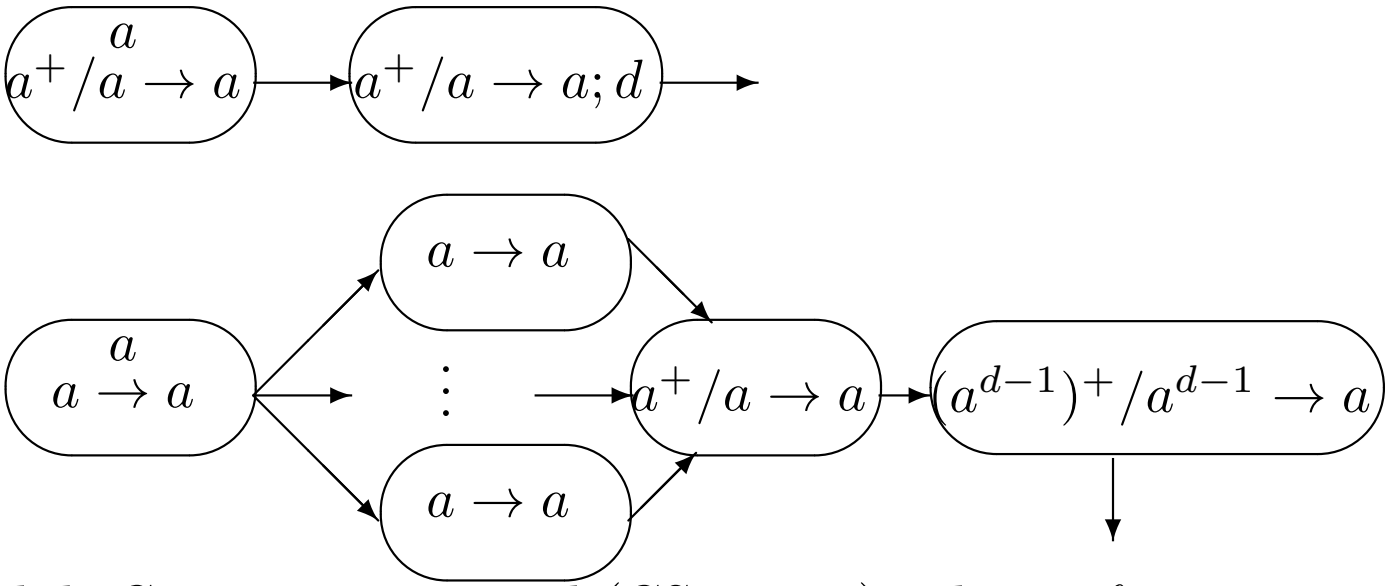
- Using matrix representation of SNP systems by Adorna, Martínez-del-Amor, Pan, Pérez-Jiménez, Zeng (BWMC 2010, CMC 2010)
- Linear algebra operations are highly parallelizeable
- SNP system and configurations as matrix and vectors, computations as vector-matrix operations
- GPU simulation speedup compared to CPU only simulation

SNP systems and Petri nets

- *"...a move of notions, tools, results in both directions, from Petri nets to SNP systems and the other way around."* - 26 Research Topics About SNP systems [Păun, BWMC 2007]
- Transforming some Petri net classes to/from SNP systems [Adorna, Cabarle, CMC 2012]
  - Preliminary work on structure and behaviour of Petri nets and SNP systems
  - Future goal: SNP systems for formal verification or analysis of systems (see next slides)

SNP with and without delays: structure and behaviour

- SNP systems with and without delays: computationally universal
- If a simple neuron (exactly one rule) has delay d, how to perform delay without using d?
  - Easy way: connect d number of simple neurons each with rule $a \to a$, in series
  - Alternative: use $d-1$ simple neurons in parallel



- Phil. Computing Journal (CSP 2012), Theory & Practice of

Behavioural equivalence in process algebra

- SNP process algebra by Barbuti, Maggiolo-Schettini, Milazzo, Tini (JLAP 2010):
  - We used SNP algebra to show behavioural equivalence of work from previous slide (WSPC, for printing 2014)
- Bisimulation of nondeterministic TP systems in weighted SNP systems (WSPC, for printing 2014)
- Another future goal: Most (or all?) P system variants are behaviourally equivalent to (a variant of) SNP systems?

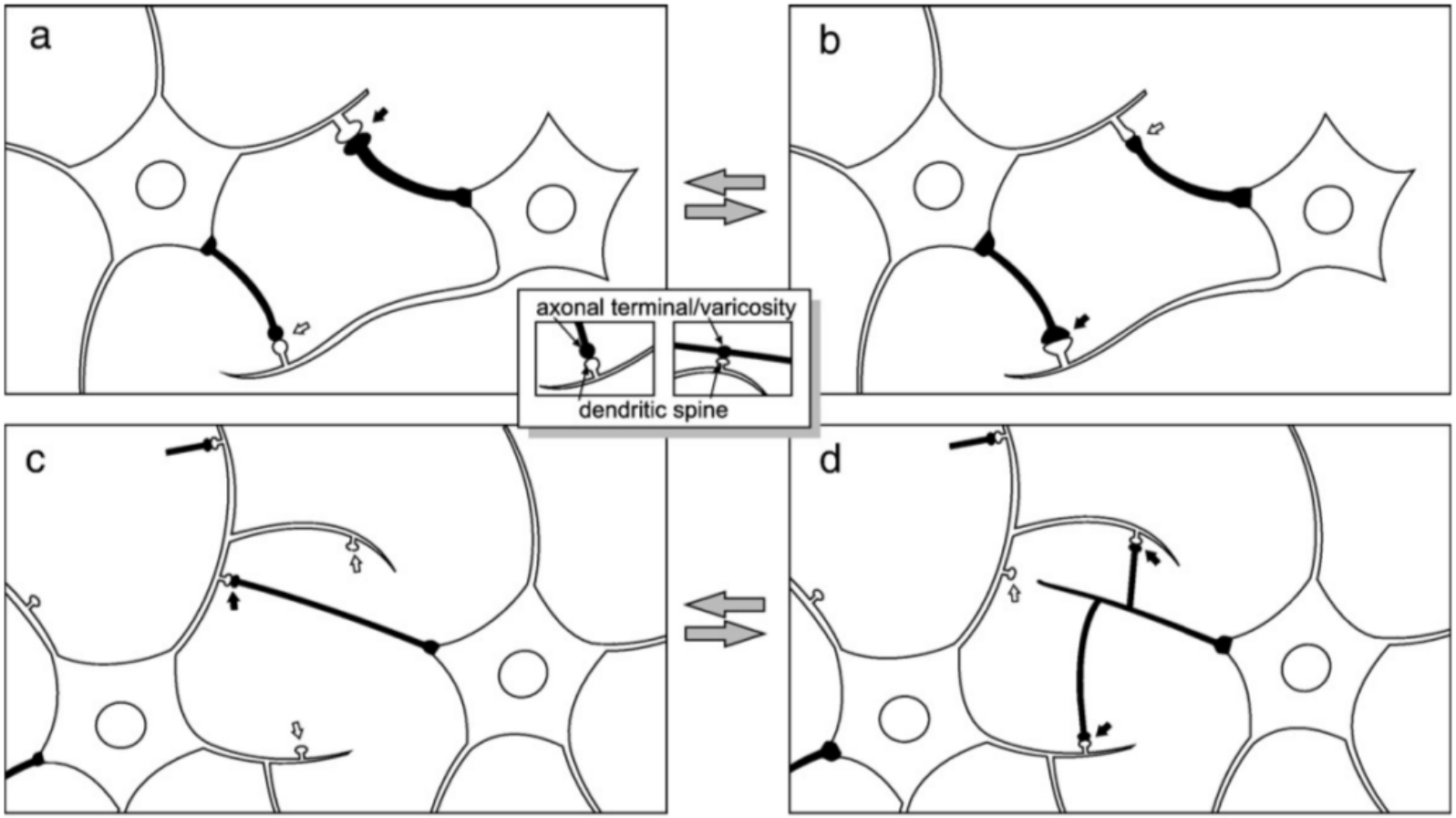
Part 3: SNP systems and structural plasticity

(Un momento!) A Biological Digression...

- Butz, M., Wörgötter, F., van Ooyen, A.: Activity-dependent structural plasticity. Brain Research Reviews 60, Elsevier (2009)
- *Functional plasticity*: change in strength of a synapse between 2 neurons
- *Structural plasticity*: change in anatomical connectivity between neurons

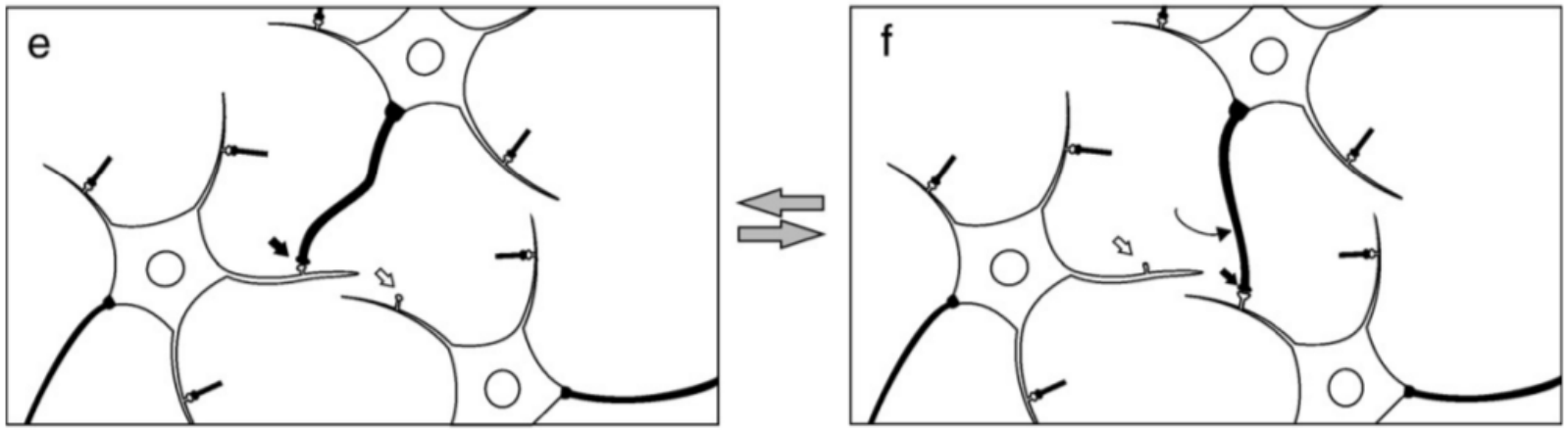
(Un momento!) A Biological Digression...

- Butz, M., Wörgötter, F., van Ooyen, A.: Activity-dependent structural plasticity. Brain Research Reviews 60, Elsevier (2009)



(Un momento!) A Biological Digression...

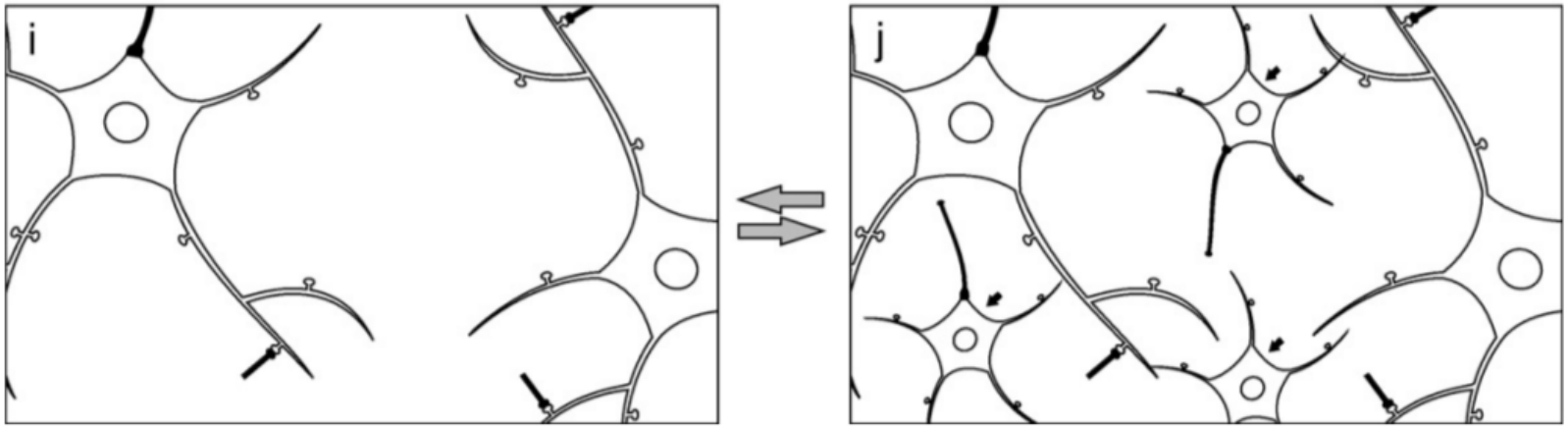
- Butz, M., Wörgötter, F., van Ooyen, A.: Activity-dependent structural plasticity. Brain Research Reviews 60, Elsevier (2009)



- *Synaptic rewiring* involves at least 3 neurons
- More complex than 2 neurons needed in functional plasticity

(Un momento!) A Biological Digression...

- Butz, M., Wörgötter, F., van Ooyen, A.: Activity-dependent structural plasticity. Brain Research Reviews 60, Elsevier (2009)



- *Neurogenesis* can be included in *synaptogenesis*

Back to SNP systems: plasticity in SNP literature

- Hebbian Learning from Spiking Neural P Systems [Gutiérrez-Naranjo, Pérez-Jiménez, WMC 2009]
  - Using functional plasticity to change synapse weight for learning
  - if spike from σ_i arrive repeatedly and shortly before σ_j fires, synapse weight of (i, j) increases
  - If spikes from σ_i arrive after the firing of σ_j, synapse weight is decreased.
  - Unlike synaptic rewiring in structural plasticity, functional plasticity only involves 2 neurons.

Back to SNP systems: plasticity in SNP literature

- SNP systems with neuron division and budding [Pan et al, SciChina 2011] and [Wang et al, CMC2010]
  - Create novel neurons (neurogenesis) using two new rules: neuron budding and neuron division rules.
  - Initial topology is changed due to new neurons and synapses (synaptogenesis)
  - Solved `SAT` using exponential neurons in linear time.
  - Synaptic rewiring or synapse deletion are not included

SNP systems: variants, features for universality

- "Standard" SNP systems [Păun, Ionescu, Yokomori (2007)]:
  - Set of neurons (nodes) and their synapses (arcs between nodes)
  - Spiking rules: $E/a^c \to a; d$, with E over $\{a\}$, $c \geq 1, d \geq 0$
  - Forgetting rules: $a^s \to \lambda$, with $s \geq 1$
  - Computationally universal in accepting and generative mode

SNP systems: variants, features for universality

- Normal forms for spiking neural P systems. [O.H. Ibarra, A. Păun, Gh. Păun, A. Rodríguez-Patón, P. Sosik, S.Woodworth (2007)]
  - without delay
  - without forgetting rules
  - simple regular expression, e.g. $a^k, k \geq 1$, or a^+
- Spiking neural P systems: Stronger normal forms. [M. García-Arnau, D. Pérez, A. Rodríguez-Patón, P. Sosik (2007)]
  - without delays and forgetting rules
  - without delays and simple regular expression
- Spiking neural P systems: An improved normal form. [L. Pan, Gh. Păun (2010)]
  - Each neuron contains at most two rules only
  - Spiking rules all have same regular expression

SNP systems: variants, features for universality

- Pan, L., Wang, J., Hoogeboom, J.H.: Spiking Neural P Systems with Astrocytes. Neural Computation (2012)
  - Without delay and forgetting rules
  - Each neuron has exactly one rule: $a^* \to a$
  - Use additional structures from neuroscience, *astrocytes*, for system "programming"
  - Astrocytes can nondeterministically remove spikes or allow them to arrive to neurons
  - Computational universality for accepting and generative SNPA

Question

- Is there a kind of SNP system that is universal if
  - Without delays and forgetting rules
  - Without additional neuroscience structures for system programming
  - Most neurons only have a simple rule: $a \to a$
  - Structural plasticity

SNP systems with structural plasticity (overview)

- without forgetting rules and delays
- Most neurons are simple (exactly one rule): $a \to a$
- Few neurons have *plasticity rules*
  - create new synapse/s
  - delete existing synapse/s
  - create (delete) then delete (create) synapse/s
- System programming: how neurons connect using synapses
  - Initial system can be a disconnected graph
- Note: system programming based on how neurons connect is not new
  - Turing, A.: Intelligent Machinery (1948)

SNP systems with structural plasticity (overview)

"...considering SN P systems with a dynamical structure. The dynamism can be achieved both in terms of neurons and synapses, or only for synapses. From birth to maturity, the brain essentially evolves at the level of synapses, learning means establishing new synapses, cutting them, making them more stable/fast when used frequently, and so on and so forth. How this can be incorporated in SN P systems?"

- 26 Research Topics About SNP systems [Păun, BWMC 2007]

SNP systems with structural plasticity

Formally, an SPSNP system is
$\Pi = (\{a\}, \sigma_1, \ldots, \sigma_m, syn, in, out)$, where:

- $\{a\}$ is the singleton alphabet (a is called spike)
- $\sigma_1, \ldots, \sigma_m$ are neurons of the form $(n_i, R_i), 1 \leq i \leq m$, with n_i indicating the initial number of spikes in σ_i and R_i is a finite rule set of two forms:
  1. Spiking rule: $E/a^c \rightarrow a$, where E is a regular expression over $\{a\}$, with $c \geq 1$;
  2. Plasticity rule: $E/a^c \rightarrow \alpha k : (i, N_j)$, where $c \geq 1$, $\alpha \in \{+, -, \pm, \mp\}$, $k \geq 1$, $1 \leq j \leq |R_i|$ and $N_j \subseteq \{1, 2, \ldots, m\}$
- $syn \subseteq \{1, 2, \ldots, m\} \times \{1, 2, \ldots, m\}$, with $(i, i) \notin syn$ for $1 \leq i \leq m$, are synapses between neurons;
- $in, out \in \{1, 2, \ldots, m\}$ indicate the input and output neurons, respectively.

Given σ_i with a plasticity rule $E/a^c \rightarrow \alpha k : (i, N_j)$,

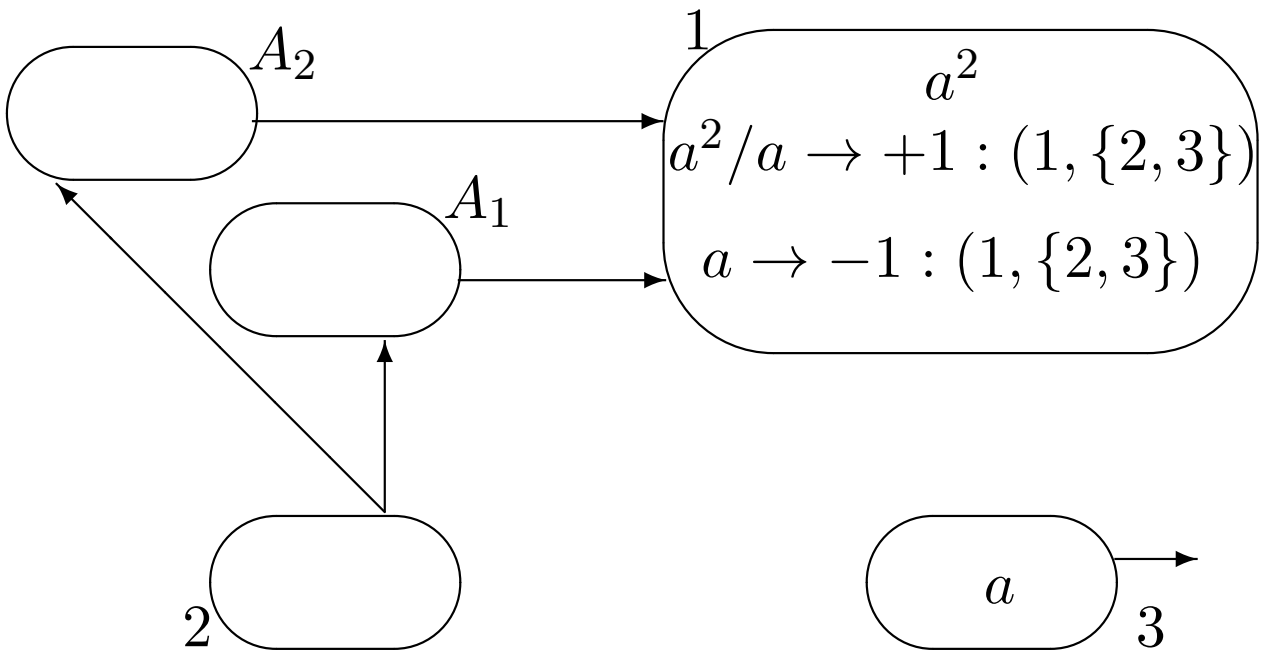
- $pres(i) = \{j|(i,j) \in syn\}$, where σ_i is the presynaptic neuron
- $pos(i) = \{j|(j,i) \in syn\}$, where σ_i is the postsynaptic neuron
- N_j is the set of neuron labels which σ_i can add (remove) synapses to (from)
- σ_i can only add or remove synapses from neurons in $pres(i) \cup N_i$, where $N_i = \bigcup N_j$

Given σ_i with plasticity rule $E/a^c \to \alpha k : (i, N_j)$,

- Depending on α:
  - $\alpha = +$, create a new synapse to k neurons in N_j; If $k < |N_j| - |pres(i)|$ then nondeterministically choose k neurons to create a synapse to, otherwise deterministically connect to all neurons in N_j;
  - $\alpha = -$, operation is similar to nondeterministic or deterministic operation when $\alpha = +$ except that k synapses to neurons in $N_j \cup pres(i)$ are removed;
  - $\alpha = \pm$ or $\alpha = \mp$ is creating (deleting) at time t, then deleting (creating) k synapses at time $t + 1$;
- When σ_i "attaches" to σ_j using a synapse, a spike is sent to σ_j in the same step
- If rule with $\alpha \in \{\pm, \mp\}$ is applied at time t, neuron is open until $t + 1$, but can only apply a new rule at $t + 2$

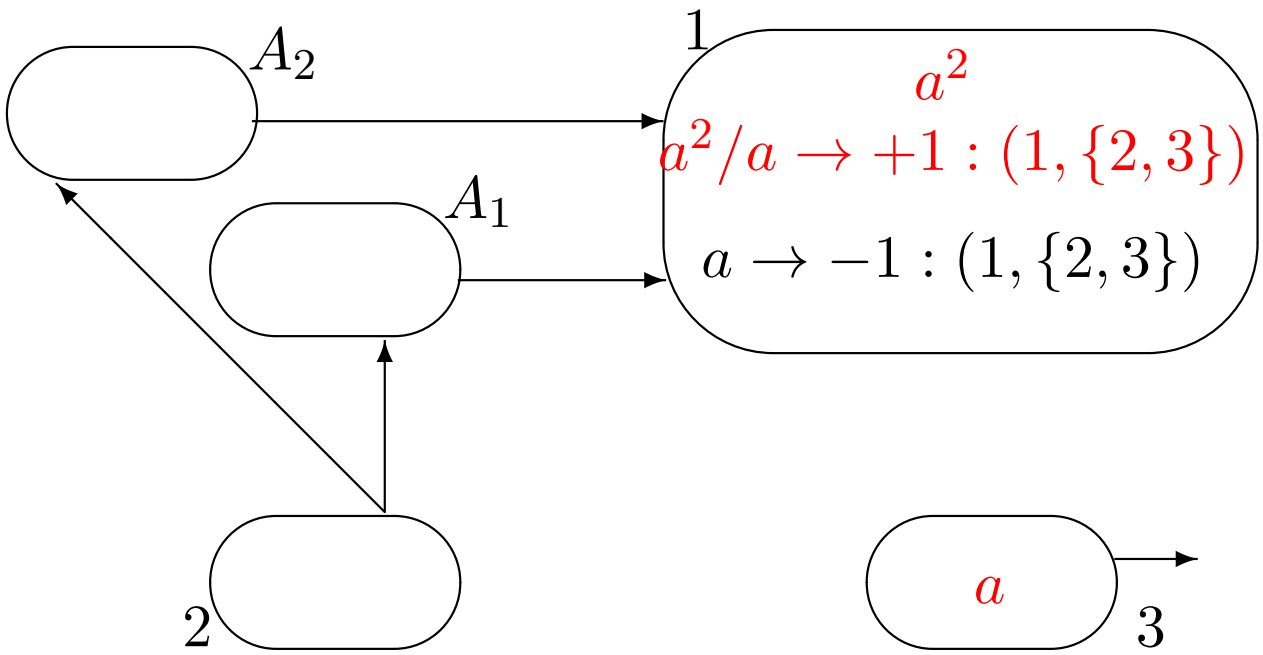
SPSNP system example: computing $\{3m+1|m \geq 0\}$

Note: we do not write $a \to a$ for simple neurons



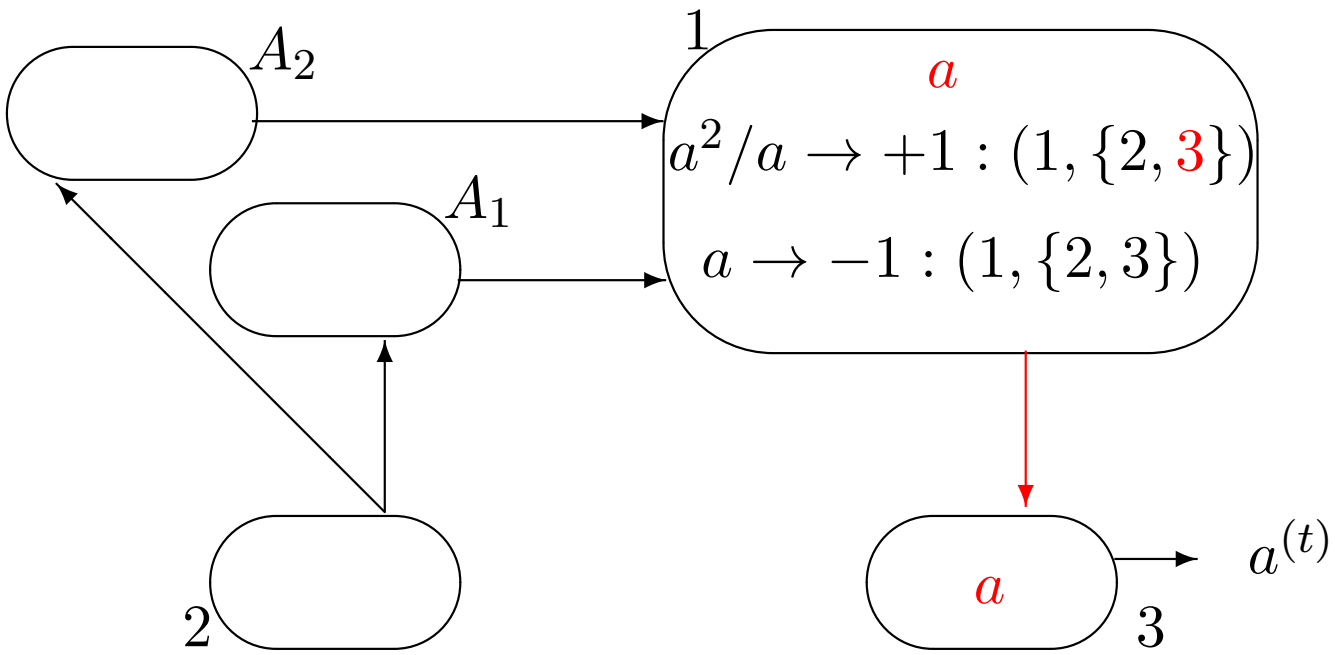
SPSNP system example: computing $\{3m+1|m \geq 0\}$

Note: we do not write $a \to a$ for simple neurons



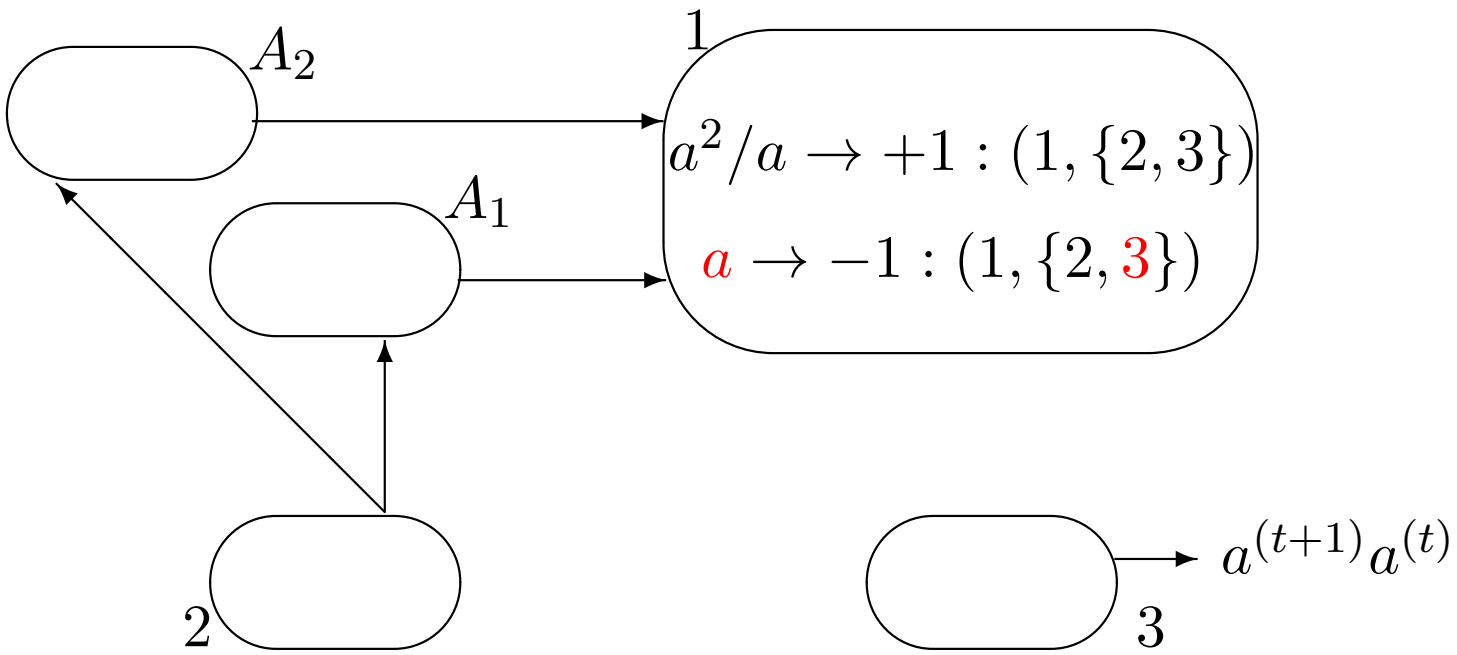
SPSNP system example: computing $\{3m+1|m \geq 0\}$

Note: we do not write $a \to a$ for simple neurons



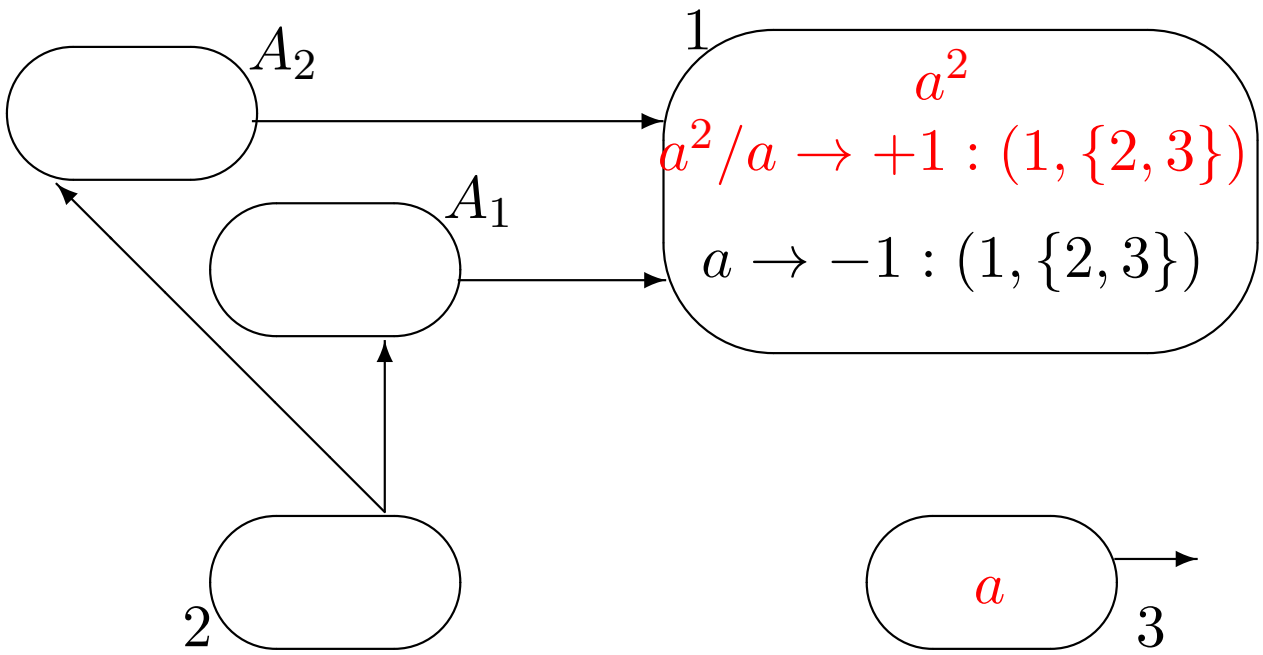
SPSNP system example: computing $\{3m+1|m \geq 0\}$

Note: we do not write $a \to a$ for simple neurons



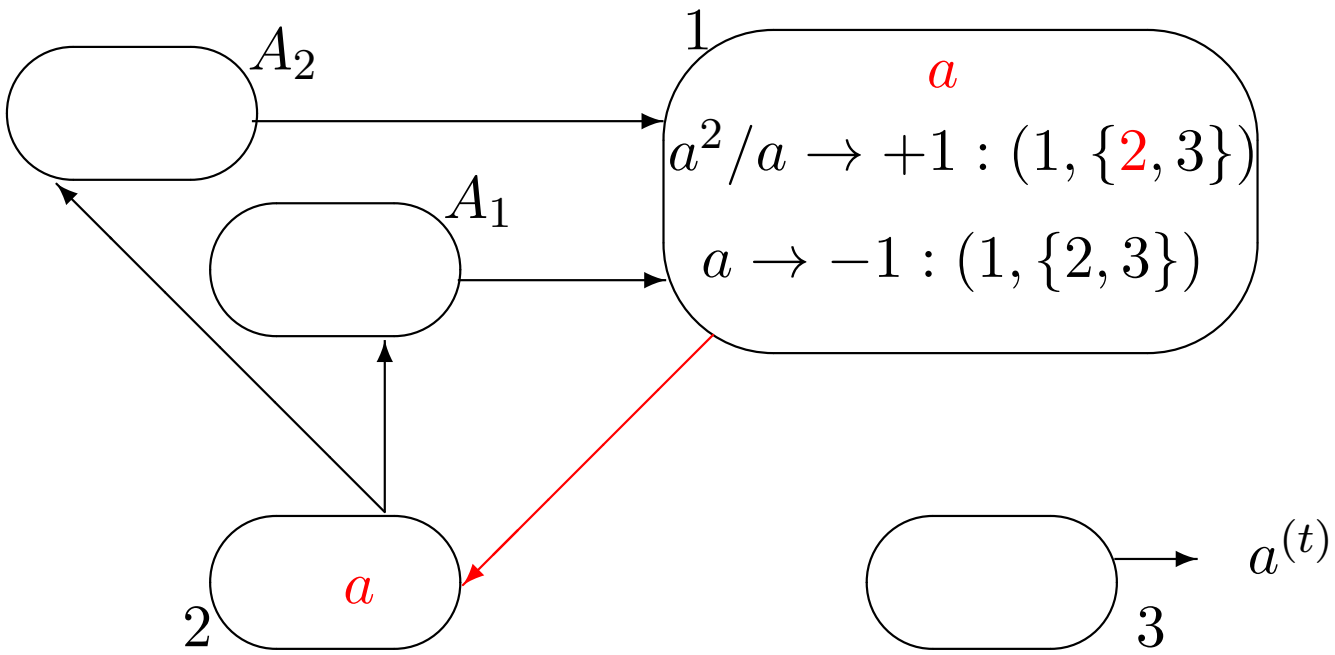
SPSNP system example: computing $\{3m+1|m \geq 0\}$

Note: we do not write $a \to a$ for simple neurons



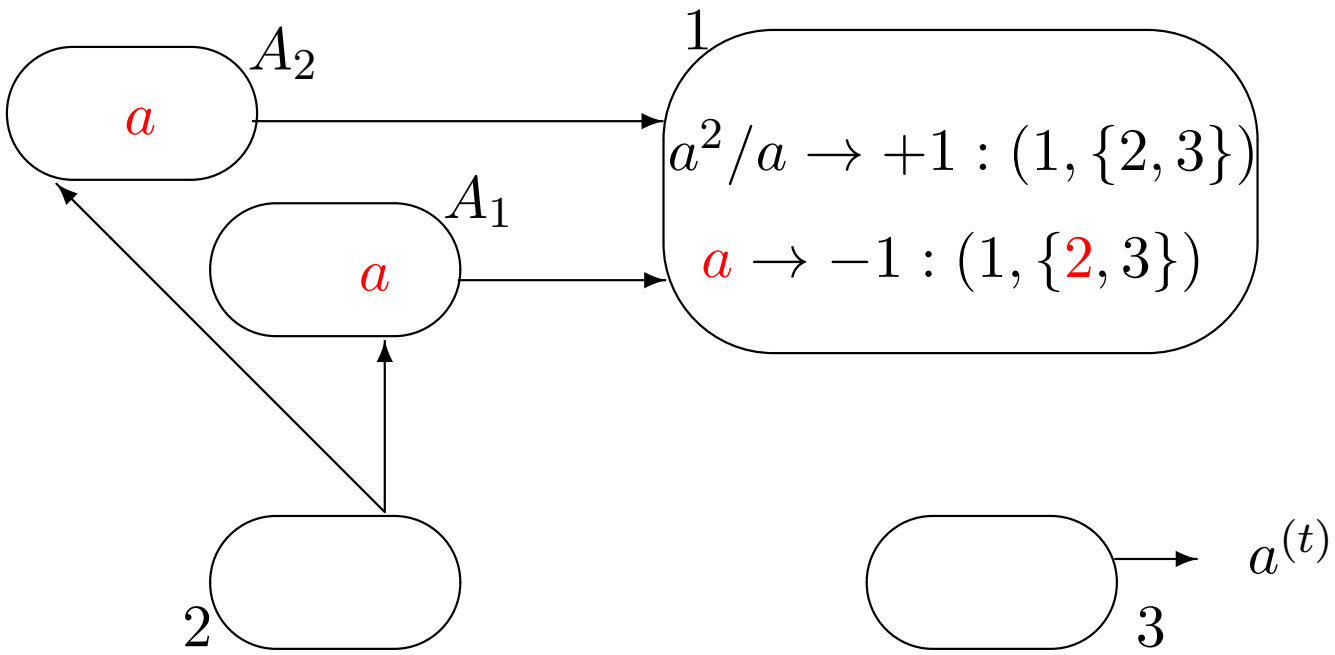
SPSNP system example: computing $\{3m+1|m \geq 0\}$

Note: we do not write $a \to a$ for simple neurons



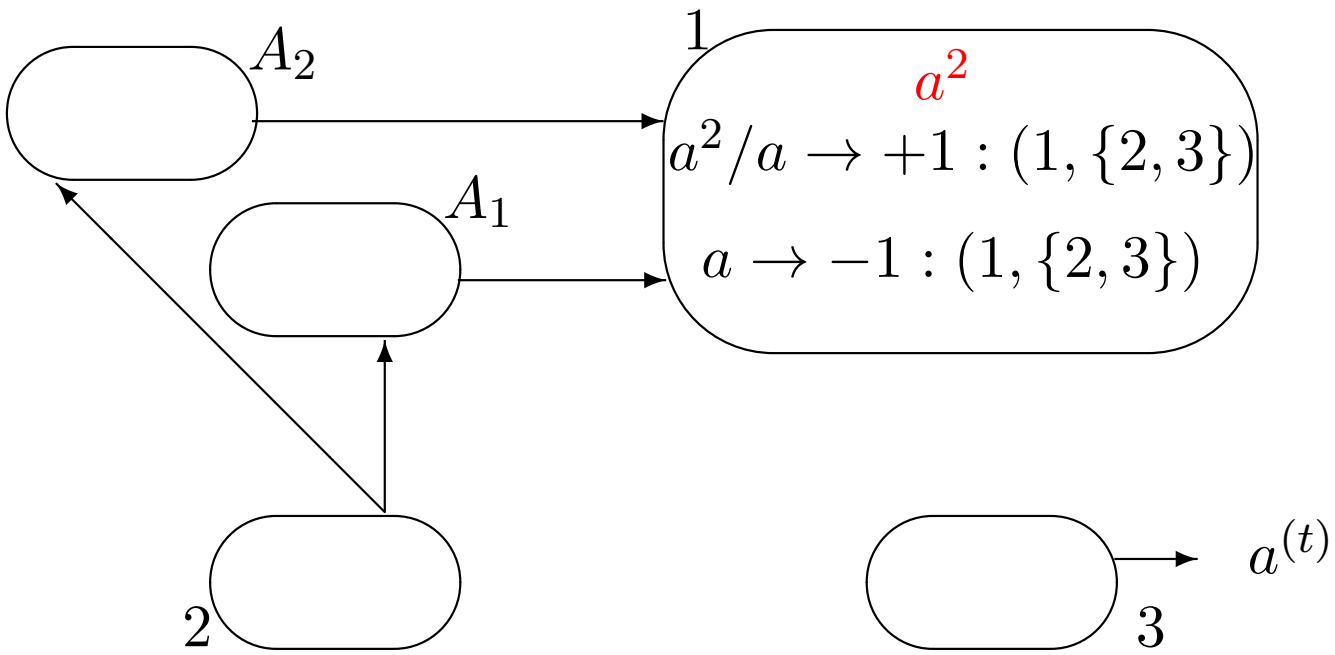
SPSNP system example: computing $\{3m+1|m \geq 0\}$

Note: we do not write $a \to a$ for simple neurons



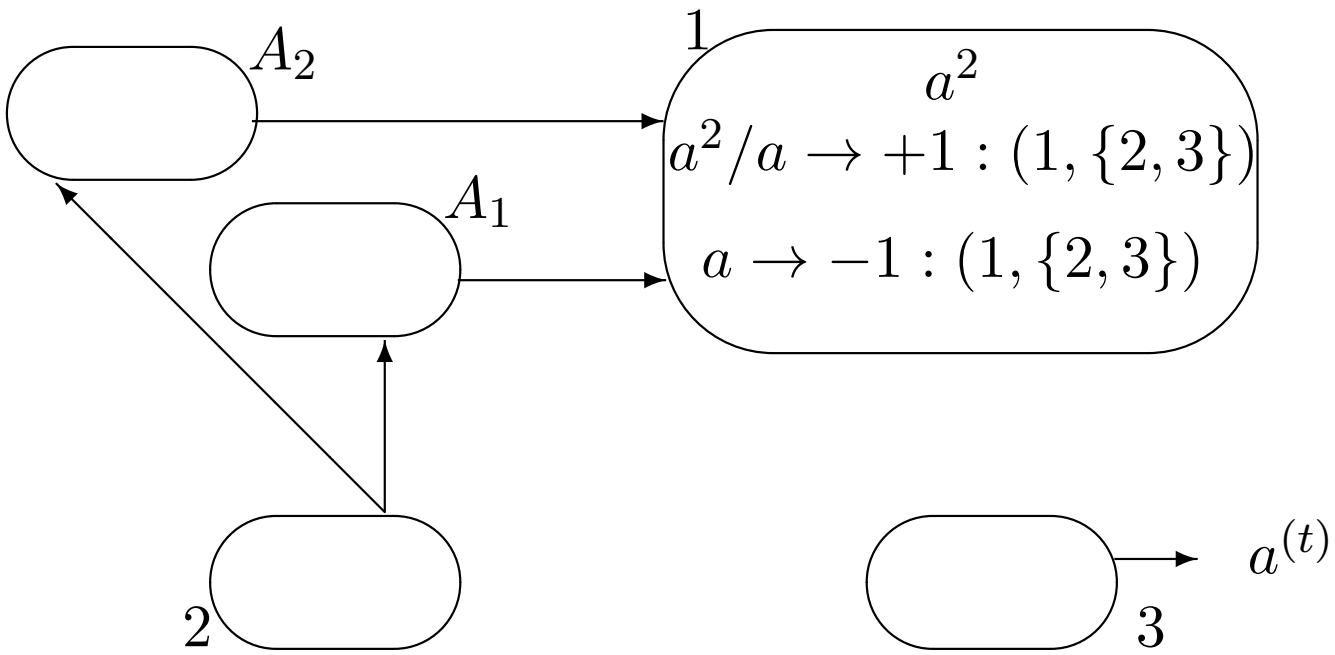
SPSNP system example: computing $\{3m+1|m \geq 0\}$

Note: we do not write $a \to a$ for simple neurons



SPSNP system example: computing $\{3m+1|m \geq 0\}$

Note: we can replace both rules of σ_1 with $a^2 \to \pm : (1, \{2, 3\})$



SPSNP system universality overview

- Generative (N_2SPSNP) and accepting ($N_{acc}SPSNP$) are universal
- Simulating register machine $M = (m, I, l_0, l_h, R)$
- Two sources of nondeterminism in SPSNP systems:
  (1) select rule to apply (rule level nondeterminism)
  (2) select neuron(s) to connect to (synapse level nondeterminism)
- Nondeterminism type (2) is sufficient for universality
- Without delays and forgetting rules
- Most neurons are simple; only "few" neurons have plasticity rules

Theorem (improved from ACMC2013)

$NRE \subseteq N_2SPSNP$.

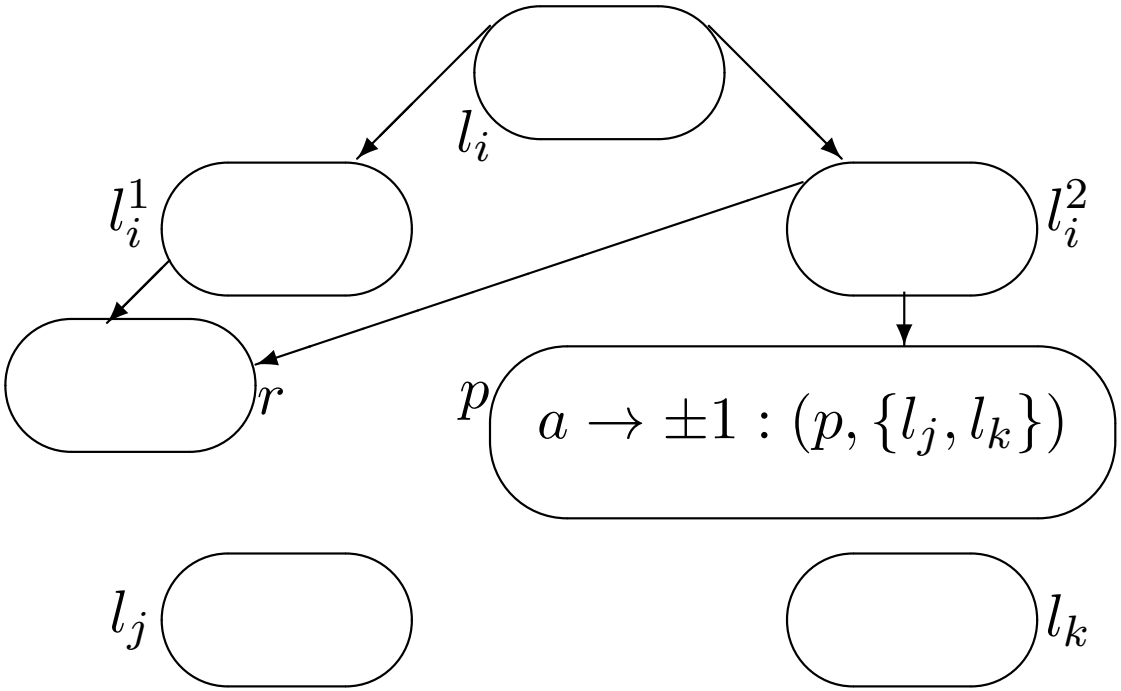


Figure : Module ADD simulating $l_i : (\texttt{ADD}(r) : l_j, l_k)$.

Theorem (improved from ACMC2013)

$NRE \subseteq N_2SPSNP$.

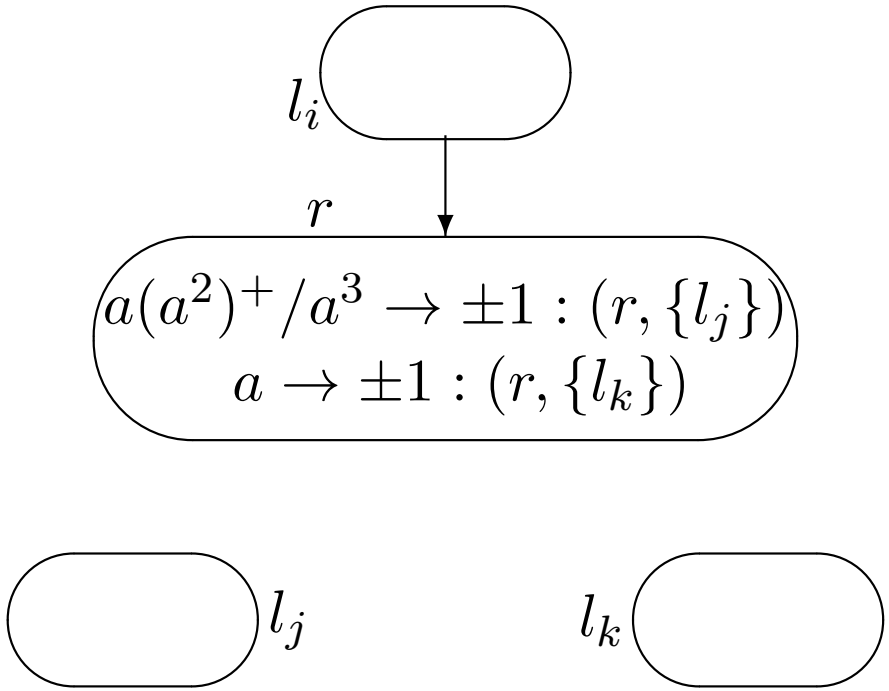


Figure : Module SUB simulating $l_i : (\texttt{SUB}(r) : l_j, l_k)$.

Theorem (improved from ACMC2013)

$NRE \subseteq N_2SPSNP$.

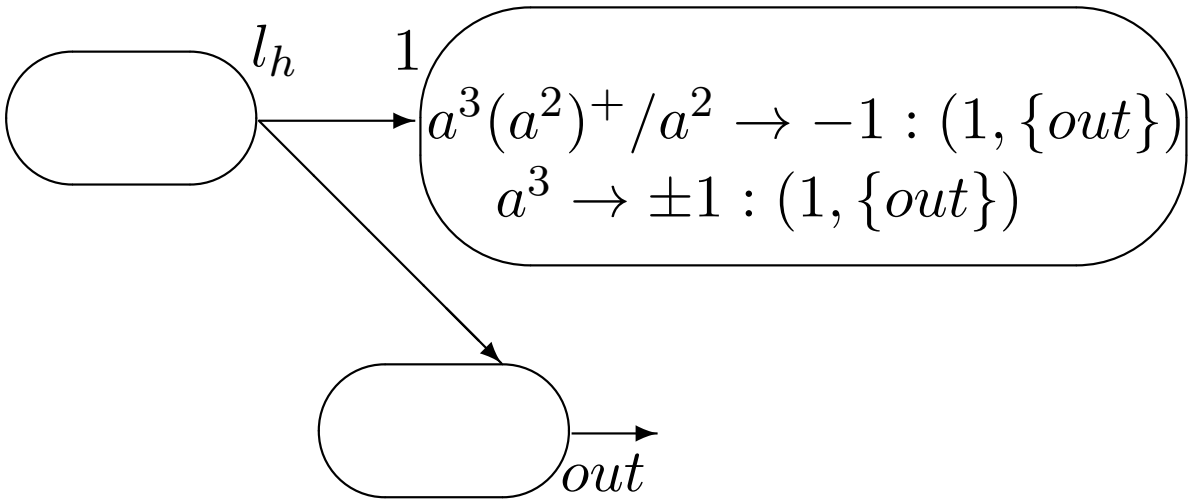


Figure : Module FIN.

Theorem (improved from ACMC2013)

$NRE \subseteq N_{acc}SPSNP$.

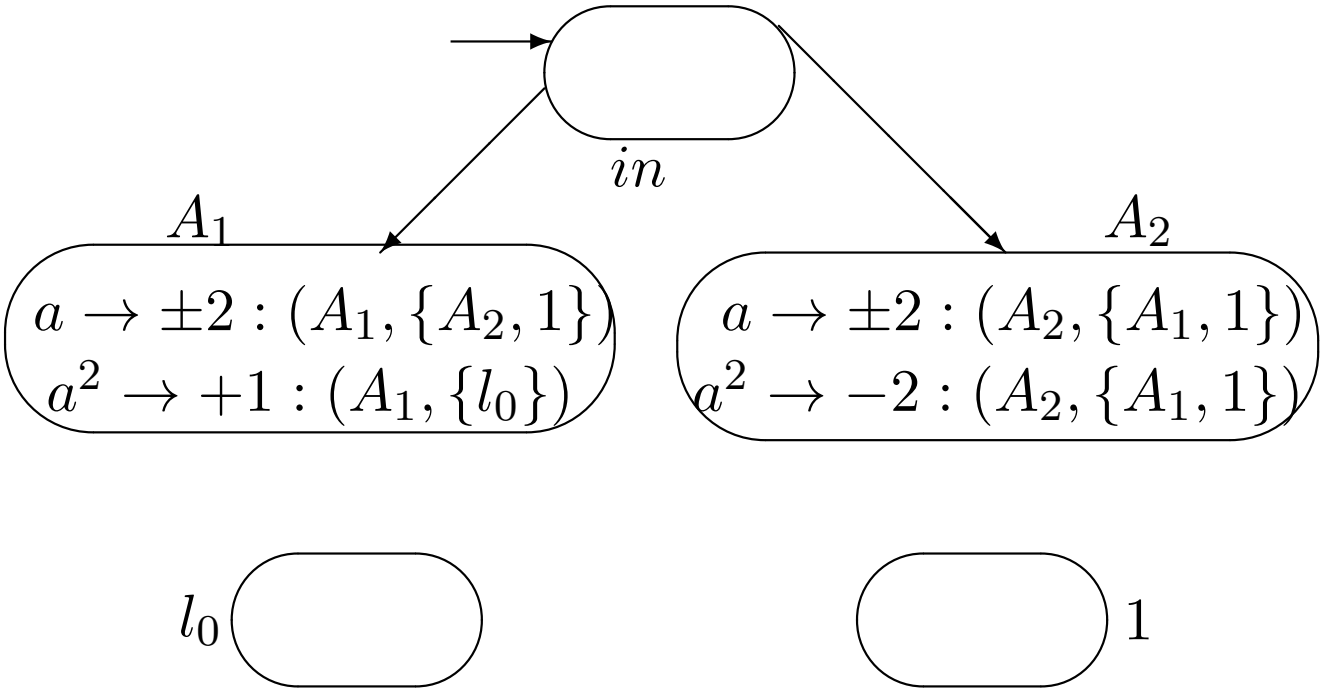


Figure : Module INPUT.

Theorem (improved from ACMC2013)

$NRE \subseteq N_{acc}SPSNP$.

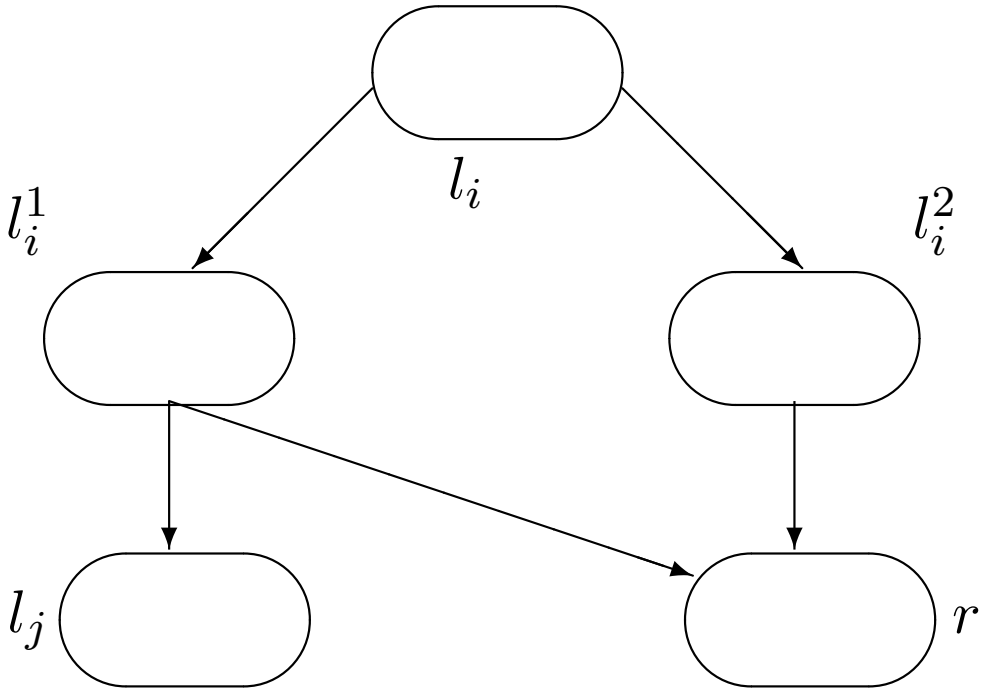


Figure : Module ADD simulating $l_i : (\texttt{ADD}(r) : l_j)$.

SPSNP system universality

- SPSNP systems are universal
  - Without forgetting rules and delays
  - With synapse level nondeterminism only
  - Most neurons are simple (with exactly one rule $a \to a$)
  - $k = 1$ for generative mode; $k \leq 2$ for accepting mode
  - $\alpha \in \{+, -, \pm\}$

Part 4: Further ideas

More on SPSNP systems

- Values for $k, |N_j|, \alpha$, can be complexity parameters for SPSNP systems
- If $\alpha \in \{\pm, \mp\}$, synapse creation and deletion are performed *sequentially*:
  - What about *parallel* creation and deletion of synapses?
  - Perform *selection* and *execution* phases similar to P systems with active membranes
  - Synapses to be created (deleted) can be the same synapses to be deleted (created) in selection phase, a "deadlock" can occur in the execution phase
- Another complexity parameter: synapse count
  - Number of synapse deletions, additions
  - In neuroscience *synaptic homeostasis*: maintaining constant number of synapses in the system
  - Synapses can be "communication channels", as in π-calculus

Again from [Păun BWMC 2007]

"...considering SN P systems with a dynamical structure. The dynamism can be achieved both in terms of neurons and synapses, or only for synapses. From birth to maturity, the brain essentially evolves at the level of synapses, learning means establishing new synapses, cutting them, making them more stable/fast when used frequently, and so on and so forth. How this can be incorporated in SN P systems?"
"A related idea is to associate a duration to each synapse (which is not of interest when the duration is constant), and to vary it in time, according to the intensity of using that synapse, and this looks rather motivated from a learning point of view."

More on SPSNP systems

- Relate SPSNP systems to dynamic graphs (networks)?
  - edge-scheduled networks and failure vulnerability [Berman, K., Networks vol 28 (1996)]
  - Evolving graphs on mobile and nomadic services (networks) [Xuan et al, IJFCS vol 14 (2002)]
- SPSNP transducers (under preparation)
  - SPSNP with both *in* and *out* neurons
  - Used to compute morphisms on (in)finite sequences
  - Extending Păun, Pérez-Jiménez, Rozenberg: Computing Morphisms by Spiking Neural P Systems. IJFCS vol 8 (2007)
- How to solve hard problems with structural plasticity?
- Structural plasticity for learning or optimization?
- Systems modeling, verification, or analysis?
- P-Lingua? GPU simulation of synapse addition/deletion/rewiring

Por fin!

Gracias por su atención!