Spiking Neural P Systems
Used as Acceptors and Transducers

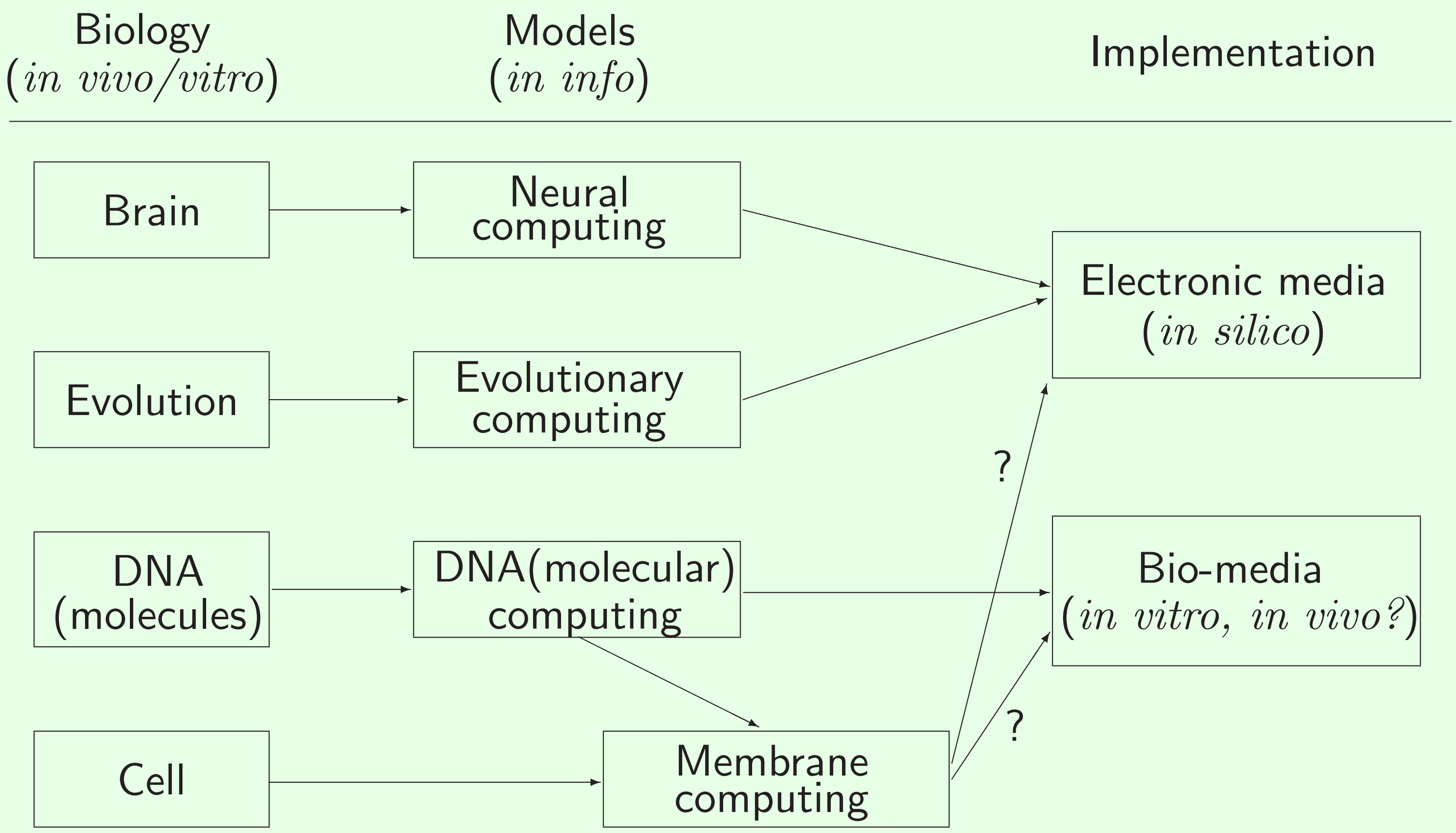
Gheorghe Păun

Romanian Academy, Bucharest
gpaun@us.es

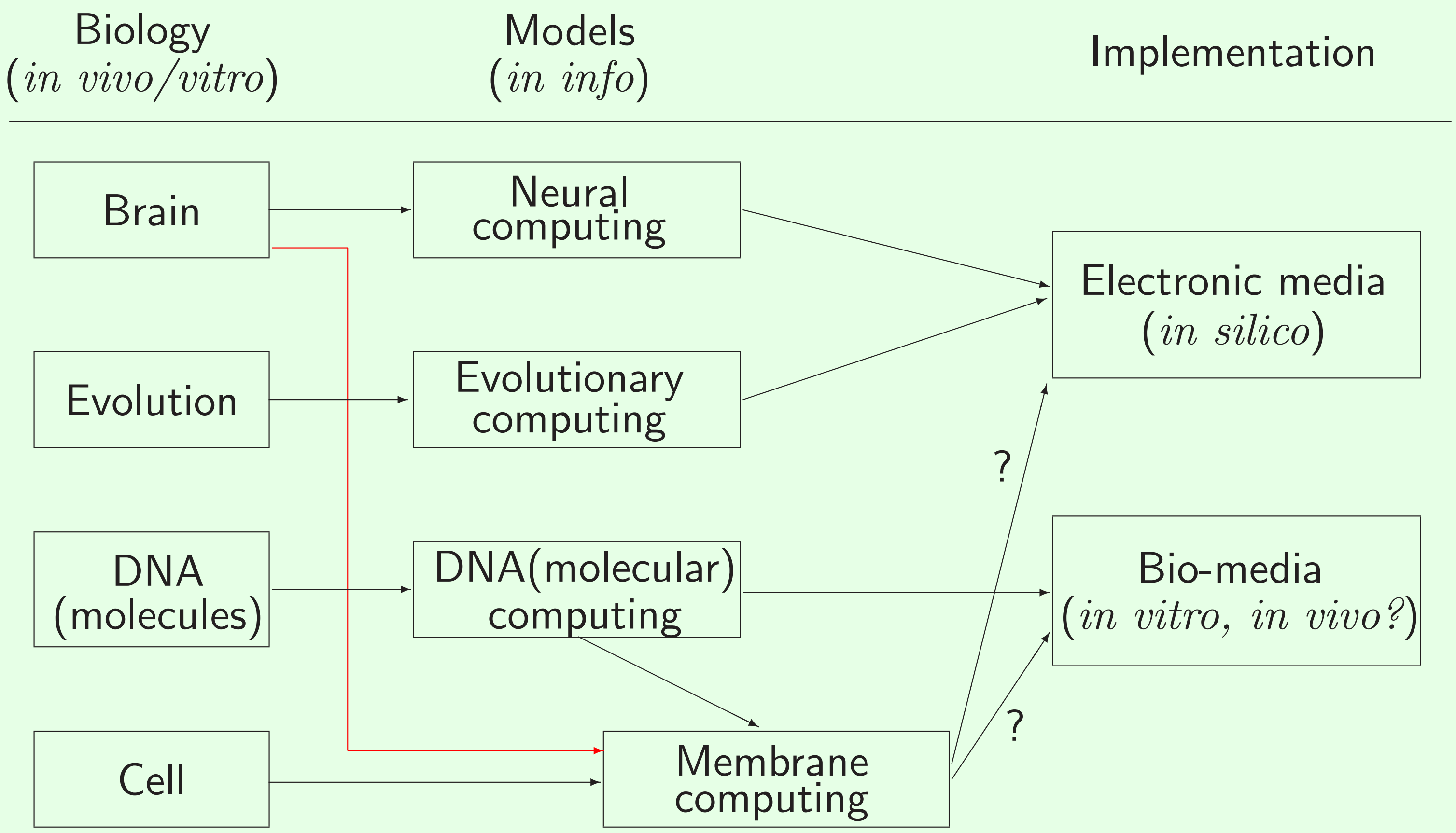
TOPICS:

- framework: natural computing/membrane computing
- generalities (spiking neurons)
- SN P systems
- types of results (generating/accepting numbers, languages)
- small universal systems
- handling strings and infinite sequences
- recent ideas: parallelism, asynchronous, complexity
- many problems and research topics

FRAMEWORK: Natural computing/membrane computing



FRAMEWORK: Natural computing/membrane computing



MEMBRANE COMPUTING:

Goal: abstracting computing models/ideas from the structure and functioning of living cells (and from their organization in tissues, organs, organisms)

hence not producing models for biologists (although, this is now a tendency)

result:

- distributed, parallel computing model

- compartmentalization by means of membranes

- basic data structure: multisets (but also strings; recently, numerical variables)

References:

- Gh. Păun, Computing with Membranes. *Journal of Computer and System Sciences*, 61, 1 (2000), 108–143, and *Turku Center for Computer Science-TUCS Report* No 208, 1998 (www.tucs.fi)
  ISI: “fast breaking paper”, “emerging research front in CS” (2003) http://esi-topics.com

- Gh. Păun, *Membrane Computing. An Introduction*, Springer, 2002

- G. Ciobanu, Gh. Păun, M.J. Pérez-Jiménez, eds., *Applications of Membrane Computing*, Springer, 2006

- Website: http://psystems.disco.unimib.it

(Yearly events: BWMC (February), WMC (summer), TAPS/WAPS (fall))

(TYPES OF) RESULTS

- Turing completeness/universality

- polynomial solutions to hard problems (time-space trade-off)

- applications: biology, bio-medicine, economics, linguistics, computer science, optimization

SOFTWARE AND APPLICATIONS:

http://www.dcs.shef.ac.uk/∼marian/PSimulatorWeb/P_Systems_applications.htm

Verona (Vincenzo Manca: vincenzo.manca@univr.it)

Sheffield (Marian Gheorghe: M.Gheorghe@dcs.shef.ac.uk)

Sevilla (Mario Pérez-Jiménez: marper@us.es)

Milano (Giancarlo Mauri: mauri@disco.unimib.it)

Nottingham, Leiden, Vienna, Evry, Iași

Spiking neural P systems

1. M. Ionescu, Gh. Păun, T. Yokomori: Spiking neural P systems, *Fundamenta Informaticae*, 71 (2006)

2. Gh. Păun, M.J. Pérez-Jiménez, G. Rozenberg: Spike trains in spiking neural P systems, *Intern. J. Found. Computer Sci.*, 17 (2006).

3. Gh. Păun, M.J. Pérez-Jiménez, G. Rozenberg: Infinite spike trains in spiking neural P systems, submitted, 2005.

4. H. Chen, R. Freund, M. Ionescu, Gh. Păun, M.J. Pérez-Jiménez: On string languages generated by spiking neural P systems, *Fundamenta Informaticae*, 75 (2007).

5. A. Păun, Gh. Păun: Small universal spiking neural P systems, *BioSystems*, to appear.

6. several other papers (e.g., 4th BWMC, WMC7, WMC8)

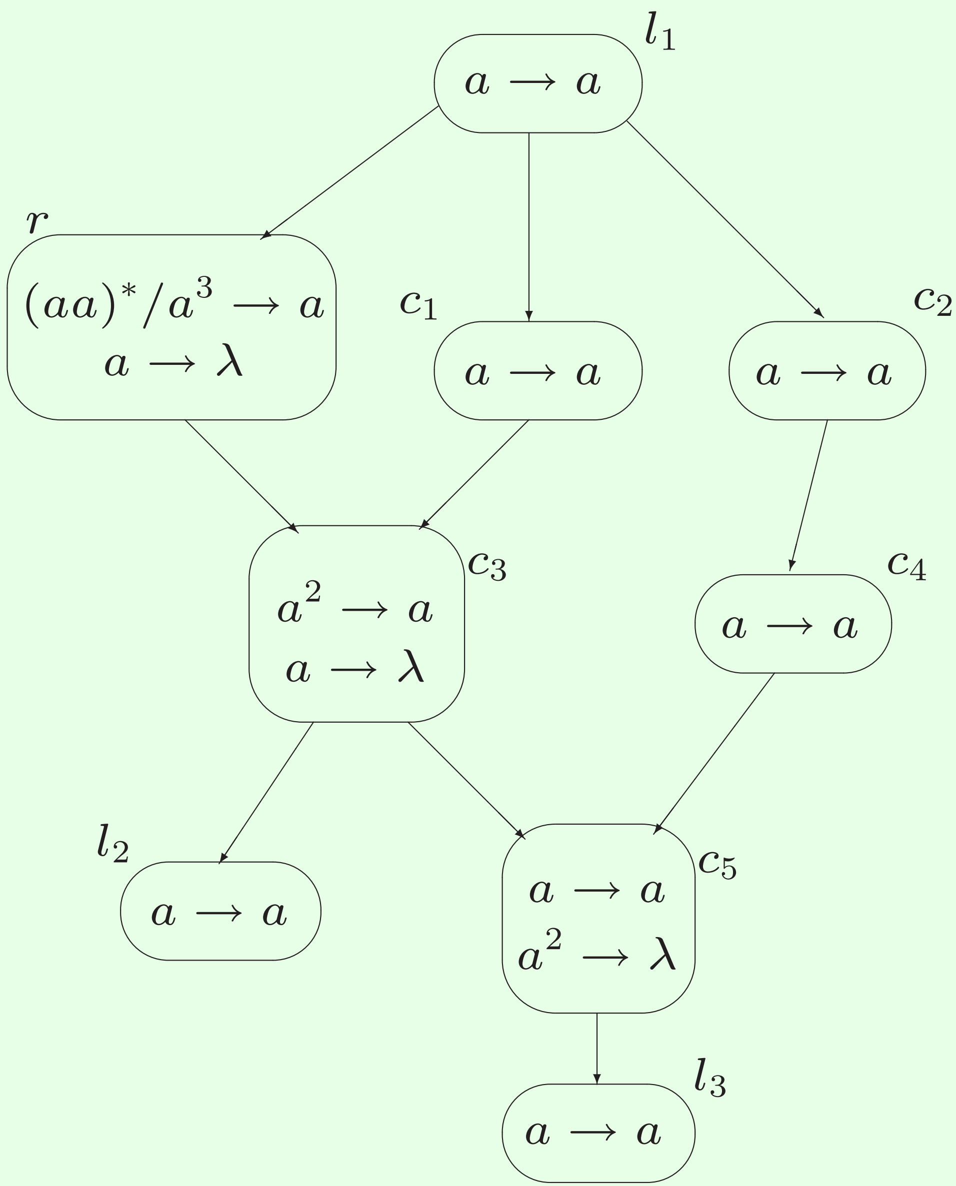
GENERAL REFERENCES ON SPIKING NEURAL NETS:

1. W. Maass: Computing with spikes. *Special Issue on Foundations of Information Processing of TELEMATIK*, 8, 1 (2002), 32–36.

2. W. Maass, C. Bishop, eds.: *Pulsed Neural Networks*, MIT Press, Cambridge, 1999.

W. Maass movie about spiking neurons:

`http://www.igi.tugraz.at/tnatschl/spike_trains_eng.html`



$$l_1 : (\mathtt{SUB}(r), l_2, l_3)$$

FORMAL DEFINITION: a *spiking neural P system* (in short, an SN P system), of degree $m \geq 1$, is a construct of the form

$$\Pi = (O, \sigma_1, \ldots, \sigma_m, syn, in, out),$$

where:

1. $O = \{a\}$ is the singleton alphabet (a is called *spike*);

2. $\sigma_1, \ldots, \sigma_m$ are *neurons*, of the form

$$\sigma_i = (n_i, R_i), 1 \leq i \leq m,$$

where:

a) $n_i \geq 0$ is the *initial number of spikes* contained by the neuron;
b) R_i is a finite set of *rules* of the following two forms:

(1) $E/a^c \to a; d$, where E is a regular expression with a the only symbol used, $c \geq 1$, and $d \geq 0$;

(2) $a^s \to \lambda$, for some $s \geq 1$, with the restriction that $a^s \in L(E)$ for no rule $E/a^c \to a; d$ of type (1) from R_i;

3. $syn \subseteq \{1, 2, \dots, m\} \times \{1, 2, \dots, m\}$ with $(i, i) \notin syn$ for $1 \leq i \leq m$ ($synapses$ among neurons);

4. $in, out \in \{1, 2, \dots, m\}$ indicate the *input* and the *output neuron*.

only out $=$ generative system

only in $=$ accepting system

both in, out $=$ computing system

Spike trains, types of output

FAMILIES: $Spik_{gen}P_m(rule_k, cons_p, forg_q)$ – generative

$Spik_{acc}P_m(rule_k, cons_p, forg_q)$ – accepting ($DSpik$, if deterministic)

Theorem 1. $NFIN = Spik_{gen}P_1(rule_*, cons_1, forg_0) = Spik_{gen}P_2(rule_*, cons_*, forg_*)$

Theorem 2. $Spik_{gen}P_*(rule_2, cons_3, forg_3) = Spik_{acc}P_*(rule_2, cons_3, forg_2) = NRE.$

Theorem 3. $SLIN_1 = Spik_{gen}P_*(rule_k, cons_p, forg_q, bound_s)$, *for all* $k \geq 3$, $q \geq 3$, $p \geq 3$, *and* $s \geq 3$.

Normal forms

PROBLEMS:

- Theorems 1, 3 for the accepting case
- Find classes of systems for which $D < ND$

  Actually, strong determinism: $L(E) \cap L(E') = \emptyset$ in each neuron
- Find classes of systems for which $SD < D$

Automata theory observation:

state complexity of E rather reduced (only a^*, a, a^2 necessary)

Language generating:

- in the standard model: over the binary alphabet

Theorem 4. (i) *There are finite languages (for instance,* $\{0^k, 10^j\}$*, for any* $k \geq 1$*,* $j \geq 0$*) which cannot be generated by any SN P system, but for any* $L \in FIN$*,* $L \subseteq B^+$*, we have* $L\{1\} \in LFSNP_1(rule_*, cons_*, forg_0)$*, and if* $L = \{x_1, x_2, \ldots, x_n\}$*, then we also have* $\{0^{i+3}x_i \mid 1 \leq i \leq n\} \in LFSNP_*(rule_*, cons_1, forg_0)$*.*

(ii) *The family of languages generated by finite SN P systems is strictly included in the family of regular languages over the binary alphabet, but for any regular language* $L \subseteq V^*$ *there is a finite SN P system* Π *and a morphism* $h : V^* \longrightarrow B^*$ *such that* $L = h^{-1}(L(\Pi))$*.*

(iii) $LSNP_*(rule_*, cons_*, forg_*) \subset REC$*, but for every alphabet* $V = \{a_1, a_2, \ldots, a_k\}$ *there are a morphism* $h_1 : (V \cup \{b, c\})^* \longrightarrow B^*$ *and a projection* $h_2 : (V \cup \{b, c\})^* \longrightarrow V^*$ *such that for each language* $L \subseteq V^*$*,* $L \in RE$*, there is an SN P system* Π *such that* $L = h_2(h_1^{-1}(L(\Pi)))$*.*

- with extended rules $(E/a^c \to a^p; d)$: over any alphabet (with $b_0 = \lambda$ or not – below, $b_0 = \lambda$)

  Theorem 5. (i) $FIN = LSN^eP_1(rule_*, cons_*, prod_*)$ *and this result is sharp, because* $LSN^eP_2(rule_2, cons_2, prod_2)$ *contains infinite languages.*

  (ii) $LSN^eP_2(rule_*, cons_*, prod_*) \subseteq REG \subset LSN^eP_3(rule_*, cons_*, prod_*)$*; the second inclusion is proper, because* $LSN^eP_3(rule_3, cons_4, prod_2)$ *contains non-regular languages; actually, the family* $LSN^eP_3(rule_3, cons_6, prod_4)$ *contains non-semilinear languages.*

  (iii) $RE = LSN^eP_*(rule_*, cons_*, prod_*)$.

The accepting case not considered yet

SMALL UNIVERSAL SN P SYSTEMS

Theorem 6. *There is a universal computing SN P system with standard rules having 84 neurons, and one with extended rules which has 49 neurons.*

Theorem 7. *There is a universal generating SN P system with standard rules having 76 neurons, and one with extended rules which has 50 neurons.*

Korec, TCS, 1996 (plus "code optimization")

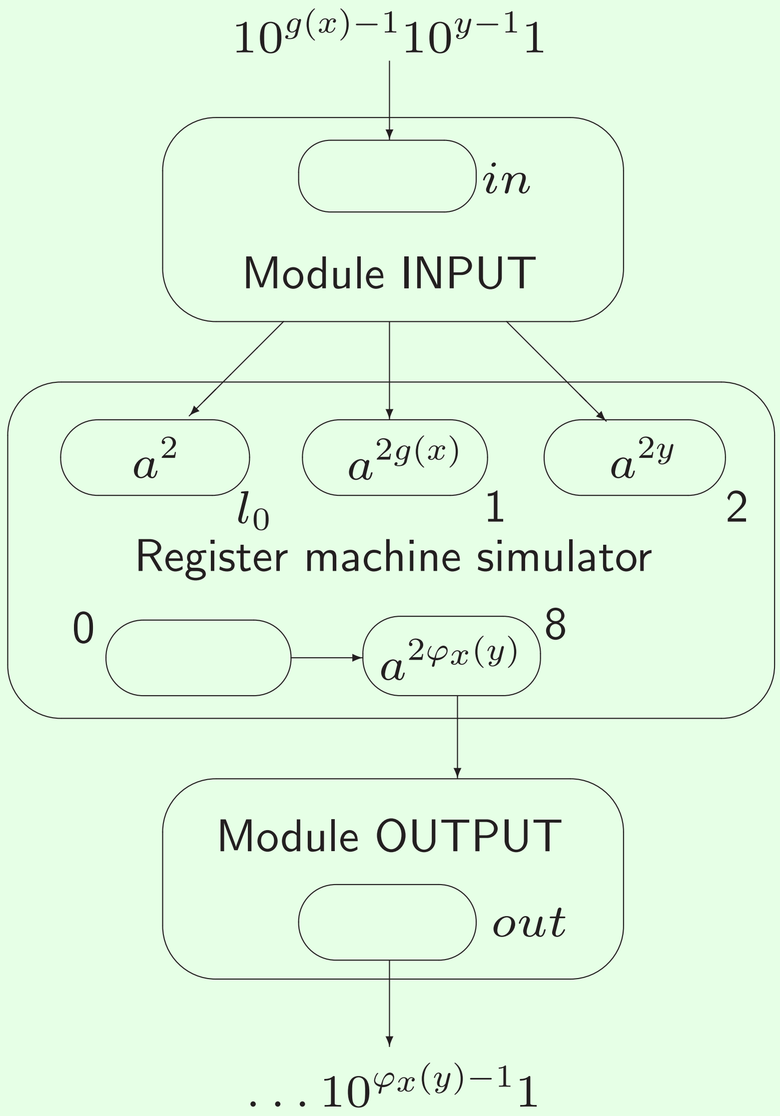


Figure 1: The general design of the universal SN P system

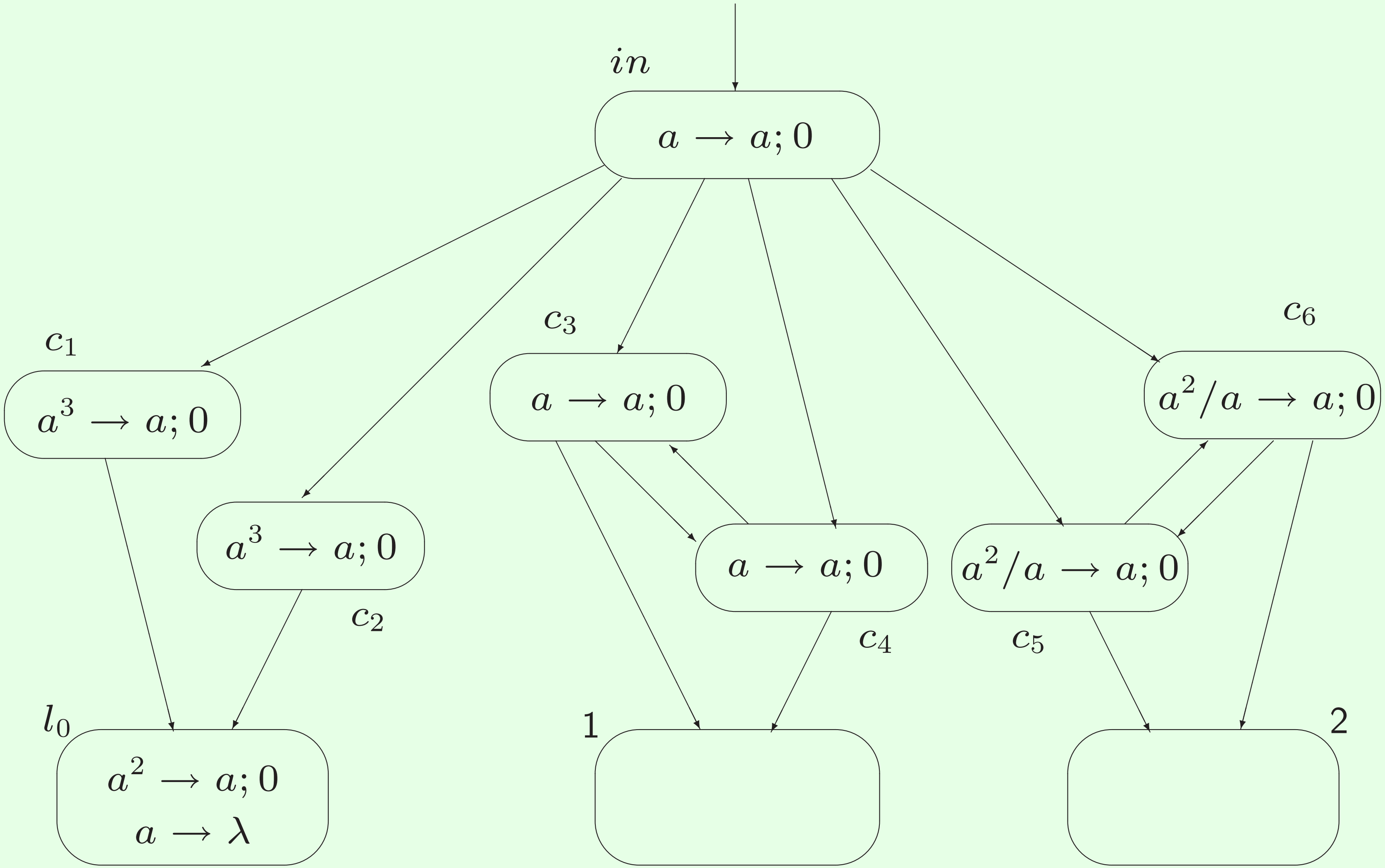


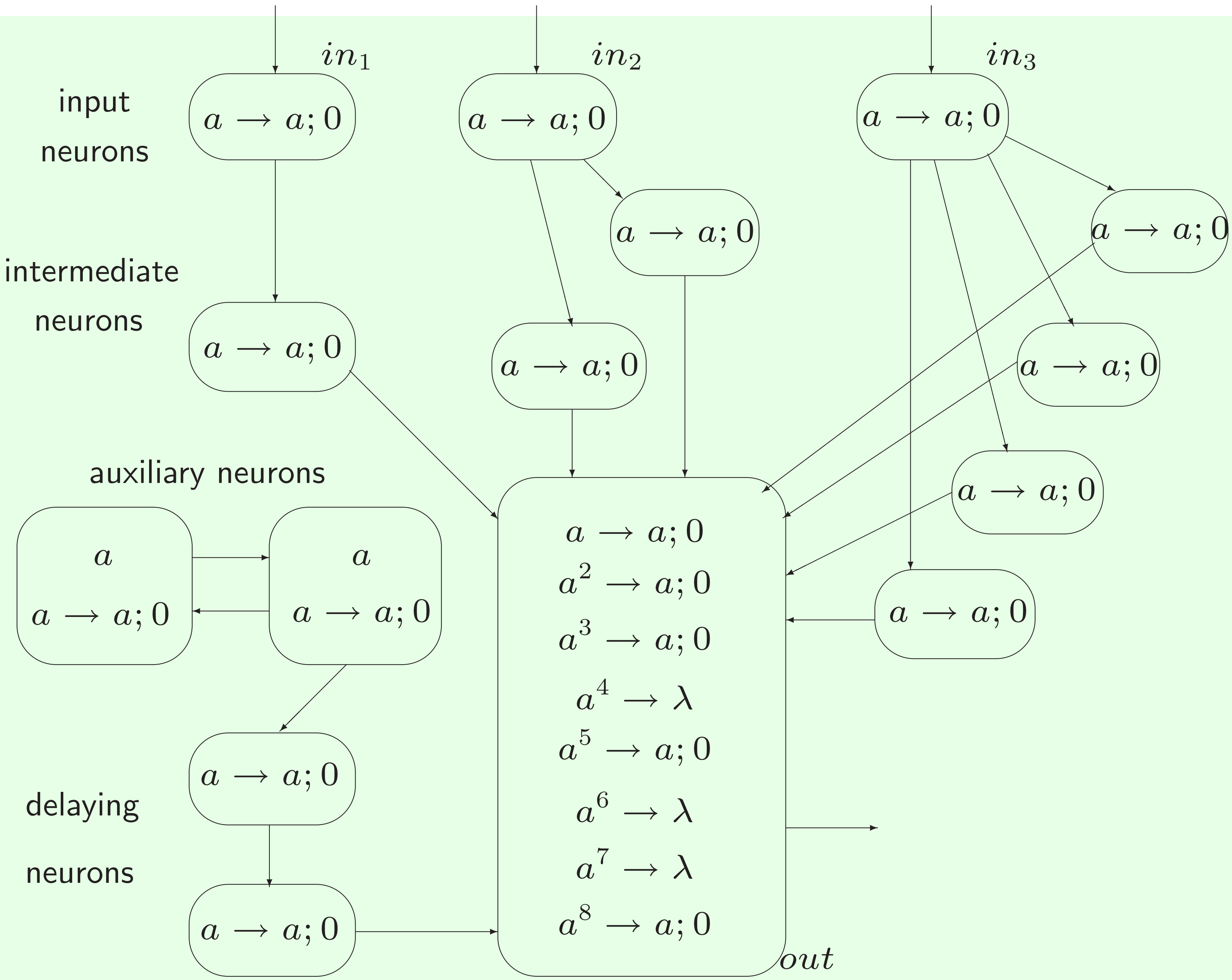
Figure 2: Module INPUT

COMPUTING (INFINITE) STRING FUNCTIONS:

Theorem 8. *Any function* $f : \{0,1\}^k \longrightarrow \{0,1\}$ *can be computed by an SN P transducer with* k *input neurons (also using further* $2^k + 4$ *neurons, one being the output one).*

Example: $f : \{0,1\}^3 \longrightarrow \{0,1\}$ defined by

$$f(b_1, b_2, b_3) = 1 \text{ iff } b_1 + b_2 + b_3 \neq 2.$$



in_1
in_2
in_3
input
neurons
intermediate
neurons
auxiliary neurons
delaying
neurons
$a \to a; 0$
a
$a^2 \to a; 0$
$a^3 \to a; 0$
$a^4 \to \lambda$
$a^5 \to a; 0$
$a^6 \to \lambda$
$a^7 \to \lambda$
$a^8 \to a; 0$
out

Computing morphisms:

- length preserving: YES (they are boolean functions)
- erasing: YES (the output is of the form w^ω)
- otherwise: YES if w^ω, NOT otherwise:

Theorem 9. *Let* $h : \{0,1\}^* \longrightarrow \{0,1\}^+$ *be a morphism with the following two properties:*
1. $|h(1)| = r \geq 2$*,*
2. we cannot write $h(0) = u^i$ *and* $h(1) = u^j$ *for some* $u \in \{0,1\}^+$ *and* $i, j \geq 1$*.*
Then, there is no SN P transducer Π *such that* $\Pi(w) = 0^s\, h(w)$ *for any given* $s \geq 0$ *and all* $w \in \{0,1\}^* \cup \{0,1\}^\omega$*.*

However, we have (a k*-block morphism* is a function $f : \{0,1\}^k \longrightarrow \{0,1\}^k$ prolonged to a function $f : \{0,1\}^\omega \longrightarrow \{0,1\}^\omega$ by

$$f(x_1x_2\ldots) = f(x_1)f(x_2)\ldots,$$

for all $x_1, x_2, \ldots \in \{0,1\}^k$):

Theorem 10. *If $f : \{0,1\}^2 \longrightarrow \{0,1\}^2$ is a 2-block morphism, then there is an SN P transducer Π such that for all $w \in \{0,1\}^\omega$ we have $\Pi(w) = 0^5 f(w)$.*

Conjecture: valid for all k

Recent developments:

- Exhaustive use of rules: universality again (both as number generators and acceptors; 3^n for n in a register)
- Asynchronous: universality for extended rules, open for usual rules (conjecture: not universal)
- Complexity: nondeterministic SN P systems solve NP-complete problems, Milano theorem for a restricted case
- Packages of spikes, specified synapses

Possible modifications:

- use the rules in the maximally parallel way
- rules $E/a^n \rightarrow a^{f(n)}$, with "easy-to-compute" partial function f; use when E covers the neuron, removing the maximal number of spikes, n, for which f is defined
- rules $E/a^\infty \rightarrow a^p$, meaning that all spikes are consumed
- inhibitory spikes, decaying time for spikes, astrocytes

Thank you!

...and please do not forget: `http://psystems.disco.unimib.it`

(with mirrors in China: `http://bmc.hust.edu.cn/psystems`,
`http://bmchust.3322.org/psystems`)