Evolving Spiking Neural P Systems

Francis G. Cabarle[2,3], Miguel A. Martínez-del-Amor[1], Xiangxiang Zeng[2], Henry Adorna[3]

[1]University of Seville.
[2]Xiamen University.
[3]University of the Philippines Diliman

Brainstorming Week on Membrane Computing
Sevilla, 2018

Evolution + “learning”

- Evolution of human brain:
  - Likely started with basic sensory-motor control.
  - *Add-on* feature: “intelligence” and “learning” likely followed this basic control.
    - Learning followed, since evolution can only do so much...
- Spiking Neural Nets (or SNNs) are generalization of previous generation NNs:
  - Asynchronous (power efficient) and non-Von Neumann (processing and memory more linked together) etc.
  - Closer to bio-reality due to spikes, e.g. for interfacing robot arms to humans.
  - Many problems, e.g. less design consensus and theory than previous generation NNs.

SN P systems so far

- Many works on evolving *parameters* (e.g. synapse weight, firing delay) and *topology*, e.g.

(1) Hebbian SN P systems.

(2) SN P systems with neuron division and budding.

(3) Optimization SN P systems.

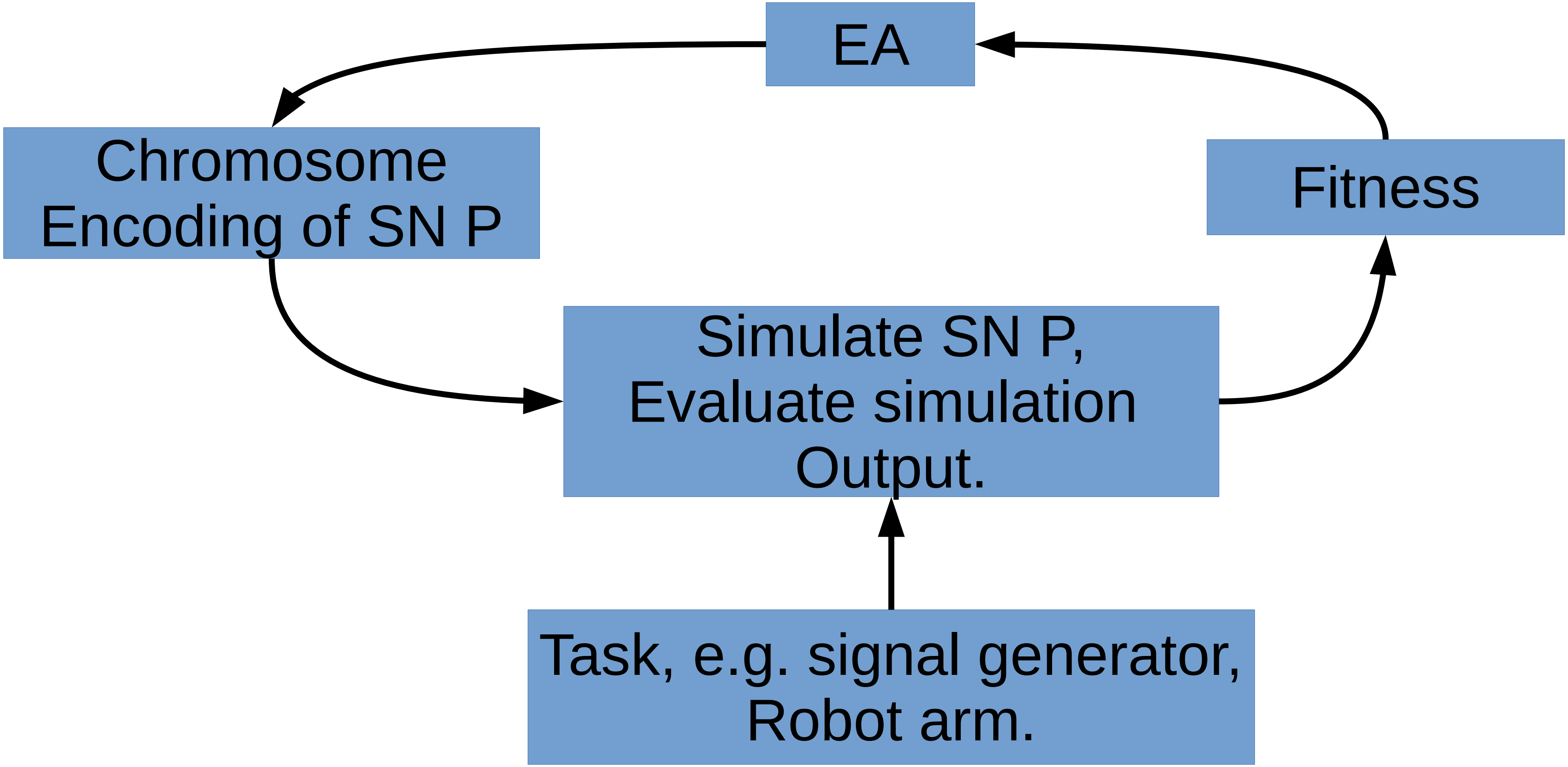
(4) SN P systems with plasticity

(5) SN P systems with scheduled synapses

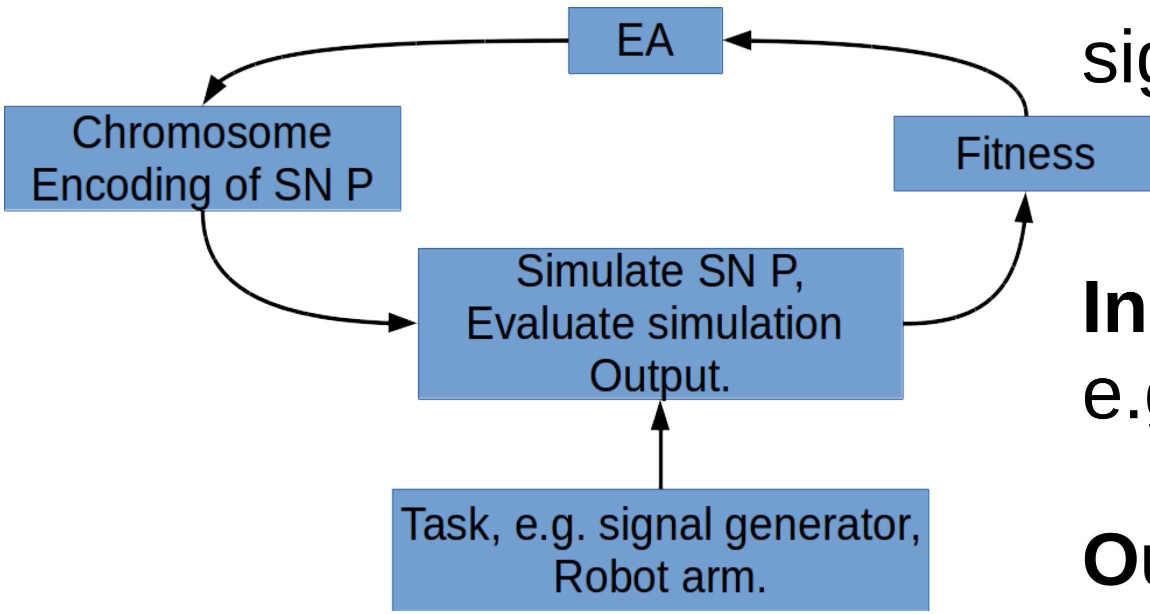
- (1), (2), (4), (5) involve “human designer”;
- (3) involves evolutionary algorithm (or EA) to evolve parameters only:
  - One view of SNN: the topology is the algorithm.

Evolving SN P systems

- “Baby steps” to evolve (some) parameters and topology: An idea.



Evolving SN P systems



Task: Spike transducer, e.g. signal generator for robots.

Input: “tonic” spike train, e.g. $a^2a^0a^2a^0a^2$...

Output: “burst” spike train, e.g. ...$a^4a^4a^0a^0a^2a^2a^0a^0a^4a^4$...

Input → **SN P parameters and topology?** → Output

Evolving SN P systems

Ideas for ***evolution paradigm***:

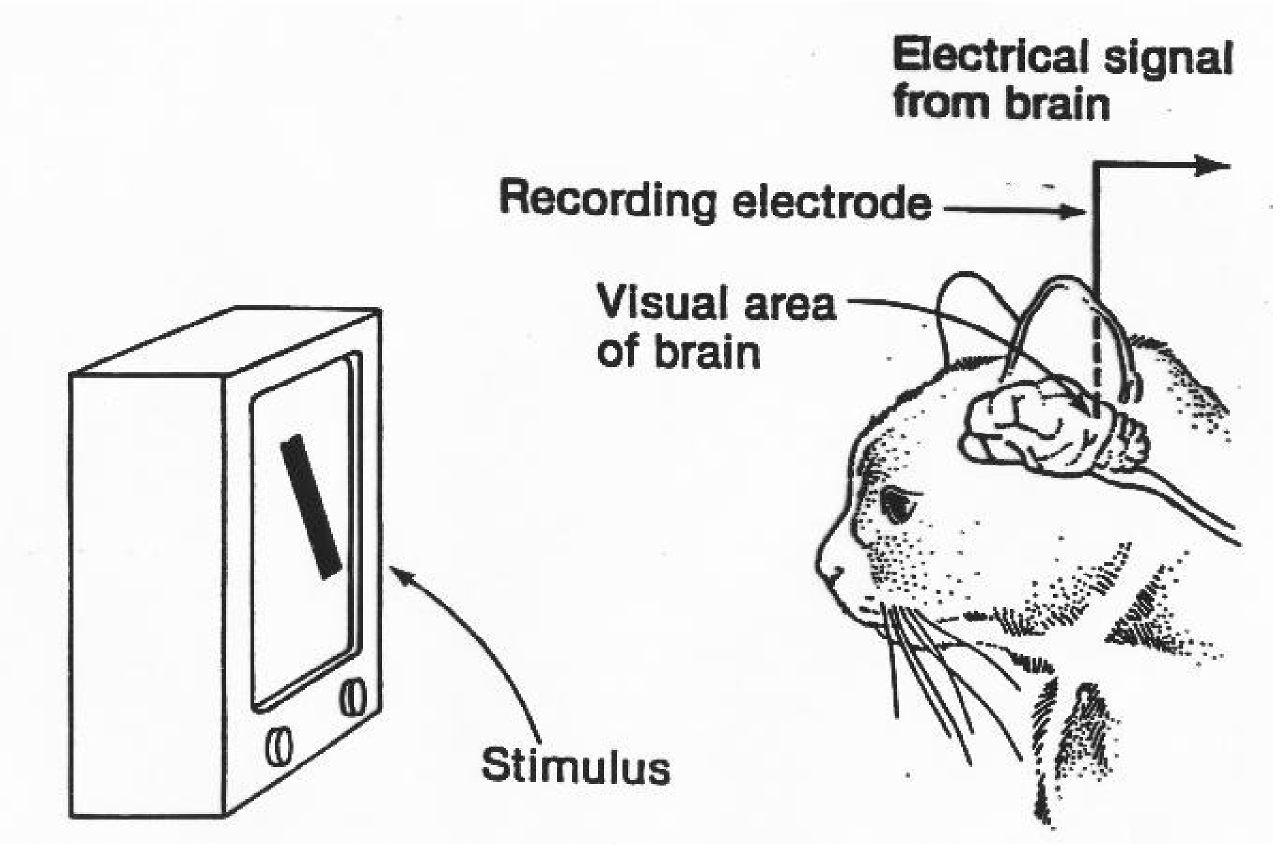
- Initially empty *adjacency matrix* of SN P:
- Proceed to *add synapses*, e.g.
  - No input neuron connects directly to an output neuron.
  - Each input neuron connects to at least one internal neuron (i.e. non-input or non-output neuron).
  - A neuron can only have either forgetting rules or spiking rules, but not both.
  - Each output neuron connects to exactly one internal neuron.
  - For feasibility, no dangling neurons, i.e. a path exists from any internal neuron to an output neuron.
- Repeat until stopping criterion is achieved.

Evolving SN P systems so far

- We have simulators, e.g. CPU, GPU, with rather efficient representations (e.g. *“sparse”* matrices, vectors).
- Some ideas for evolution of parameters and topology.
- Much work to be done, even with baby steps, e.g.
  - Experiments on hard problems, then later, real world problems.
  - Classify some problems (input-output) for some fixed parameters or topology (see evolution paradigm).
    - Response to perturbations, e.g. noise?
  - Classify some parameter or topology evolution for some fixed problems.
    - Evolve all synapses in the system? Subset?
  - Later: replacing EA with Oracle? Other ways to evolve?

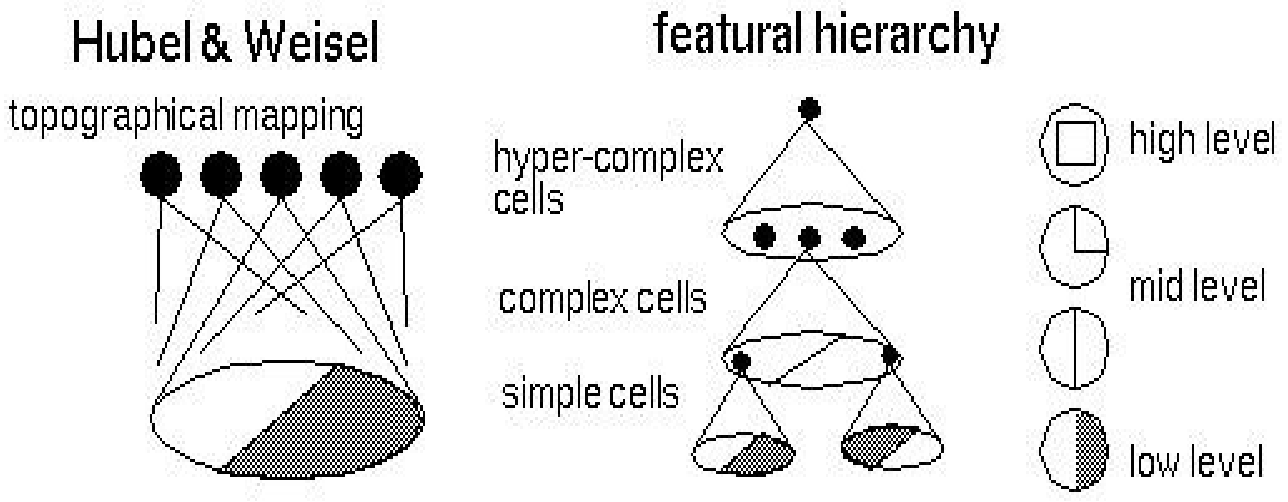
What about convolutional SN P systems?

- A bit of history:
  - [Hubel & Wiesel experiment]() 1959, 1962, 1968



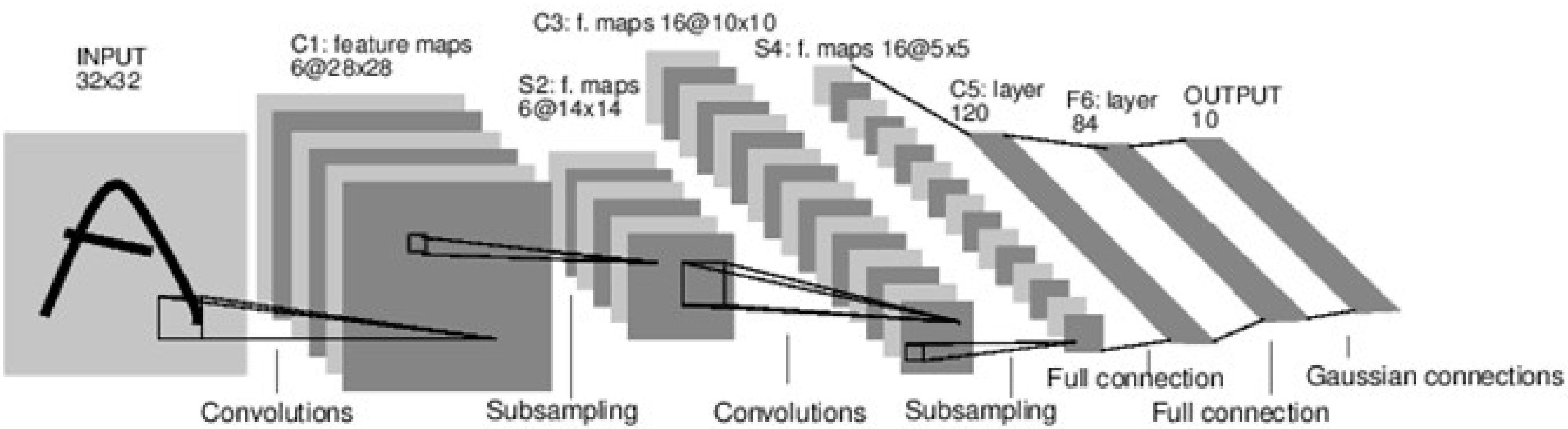
What about convolutional SN P systems?

- A bit of history:
  - Hubel & Wiesel experiment 1959, 1962, 1968



What about convolutional SN P systems?

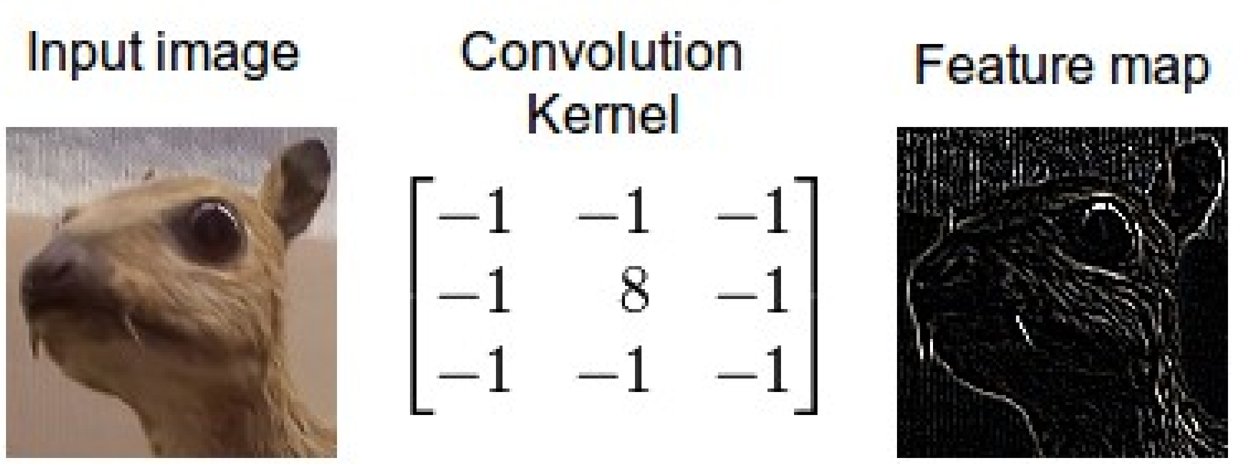
- A bit of history:
  - Le-Net 1989
  - Gradient-based learning applied to document recognition [LeCun, Bottou, Bengio, Haffner 1998]



A Full Convolutional Neural Network (LeNet)

What about convolutional SN P systems?

- A bit of history:
  - ImageNet Classification with Deep Convolutional Neural Networks [Krizhevsky, Sutskever, Hinton 2012]. Differences:
  - Use ReLu
  - It's deeper
  - Uses GPUs



What about convolutional SN P systems?

- A bit of history:

ImageNet: results for 2010-2014

Convolutional Nets galore!

- ConvNets are key in Deep Learning
  - Object detection & face detection
  - Image segmentation
  - Self-driving cars
  - Play Go & videogames
  - Speech & text processing
  - Generative: DeepDream
  - Read dreams

Lessons learned from Deep Learning

- GPUs were the **enabling technology**
  - Low-level library from NVIDIA: CuDNN
  - Specific hardware from Google came later
- Mature and end-user **tools**: TensorFlow, Caffe, Lasagne, Theano, CNTK, Keras ...
  - Large community and investment from important companies

Deep Learning: open problems

- What is going on inside the networks?
- Need too much data
- Don't extract meaning nor remember
  - Get me a knife = get me something to cut
- Supervised learning
- Need faster and more powerful machines
- (beyond deep learning and 3rd generation NN)

Thanks for the attention!

- References:

  - Schaffer, J. D. (2015). Evolving spiking neural networks: A novel growth algorithm corrects the teacher. In (CISDA), 2015 IEEE Symposium on (pp. 1-8). IEEE.
  - Gutiérrez-Naranjo, M. A., & Pérez-Jiménez, M. J. (2008). Hebbian learning from spiking neural P systems view. In WMC (pp. 217-230). Springer, Berlin, Heidelberg.
  - Pan, L., et al (2011). Spiking neural P systems with neuron division and budding. Science China Information Sciences, 54(8), 1596.
  - Zhang, G., et al. (2014). An optimization spiking neural P system for approximately solving combinatorial optimization problems. International Journal of Neural Systems, 24(05), 1440006.
  - Cabarle, Francis George C., et al. Spiking neural P systems with structural plasticity. Neural Computing and Applications 26.8 (2015): 1905-1917.
  - Cabarle, Francis George C., et al. Spiking Neural P Systems With Scheduled Synapses. IEEE Transactions on NanoBioscience 16(8) 201: 792 – 801.
  - Martínez-del-Amor, Miguel A. et al. Sparse-matrix Representation of Spiking Neural P Systems for GPU. Proc. Of 15 BWMC (2017), 161-170.
  - LeCun, Yann et al. Deep learning, Nature 521 (2015), 436-444