On the cellular automata P systems and chain reactions

Marc Barroso Mancha

Universitat de Barcelona
Email: marc.barroso4@gmail.com

This paper has been written after attending the 14th Brainstorming Week on Membrane Computing as a physics student from University of Barcelona. The work presented here tries to represent what I have learned during that period, while trying to apply some of the most innovative concepts shown in the presentations attended during that week into some interesting situations.

1 Introduction

This brief paper aims to introduce the cellular automata P systems as an example of ESNP (extended spiking neural P systems) with transmittable states, and then apply the available rules to simulate a simple model of a random walk in 2D. Let's start by formally introducing the system. We have the usual definition:

$$\Pi = (O, Q, \sigma_1, ..., \sigma_n, in, out) \tag{1}$$

· where $O = \{a\}$, such that a is called the spike
· Q is a finit alphabet of states
· σ_i are neurons, defined by: $\sigma_i = (\alpha_i, n_i, f_i, R_i)$ where
  → $\alpha_i \in$ Q, is the initial state
  → n_i is the initial number of spikes
  → f_i is the state combining function
  → R_i is a finite set of rules
· in is the input neuron
· out is the output neuron

It is also necessary to explain how the rules work. The most general form is:

$$\alpha\, /a^c \rightarrow (t_1, a^{k_1}, \beta_1), ..., (t_m, a^{k_m}, \beta_m) \text{ such that } \alpha, \beta_j \in \mathrm{Q} \tag{2}$$

that just means that when the state of the given neuron is α, and if the neuron has exactly c objects 'a' inside, it should send k_j spikes to the neuron t_j, while also transmitting the state β_j, for $1 \leq j \leq m$. When multiple rules are applied into the same neuron in the same clock tick (i.e. different states are transmitted to the same neuron σ_i), the function f_i dictates how they should be combined to obtain a unique final state.

2 Cellular automata

A cellular automata is defined as a grid of cells, such as every cell can be in one of a finite number of states. The set of cells next to each one it's called its neighborhood. Then, at every generation, some fixed rules determine the new state of each cell in terms of the current state of the cell and the states of the cells in its neighborhood. One of the most important applications is the known as "Conway's Game of Life", originally created by the mathematician John Conway in 1970. Its importance is due to the fact that it can be proven to be a universal Turing machine (that is, anything that can be computed algorithmically can be computed within Conway's Game of Life - even the Game of Life itself!). Rudolf Freund and Sergiu Ivanov show in their presentation "Extended SNP Systems with States" that the ESNP with transmittable states is analogue to a cellular automata (with the only change we are going to do is talk about neurons instead of cells). More specifically, when only two states are considered, there is an easy set of rules that enables us to simulate the Game of Life. Thus, they showed that we can obtain universality with only two states in the ESNP paradigm. The description of such system can be found on their presentation.

3 Simple nuclear chain reaction model

In order to implement these ideas into something more tangible, we can think of the following situation: we have an organized grid of atoms (represented by the neurons), where all of them are stable (in the sense that no rules could be applied initially, that is, the system would halt immediately). In the position (i, j) we introduce an unstable atom that will explode in one clock tick, releasing n number of particles (represented by the objects), that will go to any of its neighbors, making them unstable, and thus propagating some kind of state (generating the chain reaction). This is known as a two dimensional random walk. Some attractive

studies can consist of varying the number of objects an explosion yields, observing whether the reaction consume all the possible atoms or not (if an atom that has already exploded is considered to be destroyed insted of being replaced), or considering some time of interaction in every step. Another interesting behavior (also more difficult to implement) would be trying to change the geometry of the system (so the number of neighbors could vary from atom to atom), and see if you could get more efficiency some way or another.

This is obviously the first iteration of the model one can think of: further complications can be considered, as making the grid in 3D (just by adding layers upon layers of atoms), or even trying to add probabilities so further atoms than the more direct neighbors have a chance of becoming unstable. This model can be used to simulate the path of a photon that emerges from the Sun's core and is trying to reach the surface. A very simple model has been implemented at what aims to be an ESNP simulator, programmed in Python. It consists of a square grid of arbitrary dimensions, and an unstable atom in the middle. The initial reaction lasts two ticks, and send one spike in two random different directions. For the sake of simplicity, we have used three different states: 0 - stable atom, 1 - unstable atom, 2 - already exploded atom, even if two were already enough, as explained before. We understand that this model has some difficulties (we only used the most direct neighbors; this can be extended to have probabilities for all 8 adjoin neurons), and can even be too simplified (we know that atoms won't be organized in a rectangular grid, as shown here, but will have some kind of spatial distribution). But while modeling this problem, we found out that this could also be implemented as some kind of A^* path-finding algorithm or some kind of path algorithm, only halting the computation when a certain point is reached or if no rules have been applied, and then letting a lot of systems run in parallel. With a comparison between the number of rules used, one can actually get a good representation of the optimal route. Could we explot from the fact that we can acces more states and take advantage of it, working under the P-systems paradigm? Further research and modeling must be done to answer this.