A Continuous-Time Spiking Neural Network Paradigm

Alessandro Cristini*, Mario Salerno, and Gianluca Susi

Departement of Electronic Engineering, University of Rome “Tor Vergata”,
via del Politecnico 1, 00133 Rome, Italy
alessandro.cristini84@gmail.com,
{salerno,gianluca.susi}@uniroma2.it

Abstract. In this work, a novel continuous-time spiking neural network paradigm is presented. Indeed, because of a neuron can fire at any given time, this kind of approach is necessary. For the purpose of developing a simulation tool having such a property, an ad-hoc event-driven method is implemented. A simplified neuron model is introduced with characteristics similar to the classic Leaky Integrate-and-Fire model, but including the spike latency effect. The latency takes into account that the firing of a given neuron is not instantaneous, but occurs after a continuous-time delay. Both excitatory and inhibitory neurons are considered, and simple synaptic plasticity rules are modeled. Nevetheless the chance to customize the network topology, an example with Cellular Neural Network (CNN)-like connections is presented, and some interesting global effects emerging from the simulations are reported.

Keywords: Neuron Model, Spike Latency, Spiking Neural Network, Synaptic Plasticity, Continuous-Time Paradigm, Event-Driven Simulation.

1 Introduction

In recent decades there has been a significant increase in the development, implementation and general purpose use of spiking neural networks [1,2,3]. The attractiveness of this kind of neural networks lies in the bio-inspired neuron models and the related peculiar characteristics, such as: subthreshold decay of the membrane potential, spatio-temporal integration of the incoming synaptic inputs, excitatory and inhibitory effects, threshold phenomena, spike latency, synaptic plasticity, etc. Several neuron models have been proposed in the literature, from the simplest Integrate-and-Fire [4] to the most bio-realistic Hodgkin-Huxley models [5]. However, these are typically described by means of ODEs (Ordinary Differential Equations). Usually, the more ODEs are complex, the more the neuronal membrane potential is faithfully followed. A comparative

[*] Corresponding author.

S. Bassis et al. (eds.), *Recent Advances of Neural Networks Models and Applications*,
Smart Innovation, Systems and Technologies 37, DOI: 10.1007/978-3-319-18164-6_6

review of the main neuron models is listed in [6]. With the aim of making possible simulations of very large networks on a PC (Personal Computer), often the choice fall on the simplest model. Indeed, the latter allows the investigation of global effects arising only in the case of large networks of spiking neurons, such as: Polychronization [7,8] or the formation of spontaneous neuronal groups that can be affected by the Neuronal Group Selection [9,10]. Moreover, in some cases I&F model can be analytically studied (e.g., [11,12]).

There are two main methods to simulate such networks: clock-driven (or "synchronous") and event-driven (or "asynchronous") strategies. In the first one, all neurons are updated simultaneously at every tick of a clock, whereas in the second one, all neurons are updated only when they receive or emit a spike. The latter kind of strategies is developed for exact simulations, thus allowing a high precision computation. The obvious drawback of clock-driven methods is that spike timings are aligned to a grid (ticks of the clock), thus the simulation is approximated even when the differential equations are computed exactly. Other specific errors come from the fact that threshold conditions are checked only at the ticks of the clock, implying that some spikes might be missed. Whereas, event-driven methods implicitly assume that we can calculate the state of a neuron at any given time, i.e., we have an explicit solution of the differential equations [13]. This condition cannot be always satisfied; for instance, the Hodgkin-Huxley equations have no explicit solution.

However, in the present work we apply an event-driven method in order to implement a novel continuous-time spiking neural network paradigm. The neuron model is not described using ODEs, but it explicitly makes an algebraic sum of any incoming inputs to a target neuron. Indeed, both excitatory and inhibitory effects are considered. Each input is weighted by a synaptic strength, that may be affected by the synaptic plasticity (see [14] for an overview about the biological mechanisms), thus we have implemented simple rules (described in *synaptic plasticity rules* section) to take into account this phenomenon. Note that, in this model when a threshold is crossed, the neuron will fire with a continuous-time delay called latency [15]. Of course, the threshold crossing is often prevented by the subthreshold decay. Therefore, a spatio-temporal integration is implemented. Finally, after the spike generation, the neuron will be reset to its resting potential, becoming not excitable for a time equal to the absolute refractory period. Also, we have considered a Cellular Neural Network (CNN)-*like* topology [16], in order to arrange the connections among neurons; then, each neuron fires to a number of target neurons belonging to a fixed neighborhood. In *simulation results* section, we will show simulation results about global effects arising thanks to this kind of model and neuronal paradigm.

2 Neuron Model

We propose here a simplified neuron model in which the variables are updated step-by-step (i.e., in correspondence to incoming events). Note that, only normalized real quantities are considered. In addition, we call "firing neuron" an emitting neuron, and "burning neuron" a receiving one (see Fig. 1). Furthermore, we define "passive mode" the operating mode of the neuron when its inner state is less than a threshold, "active mode" otherwise.

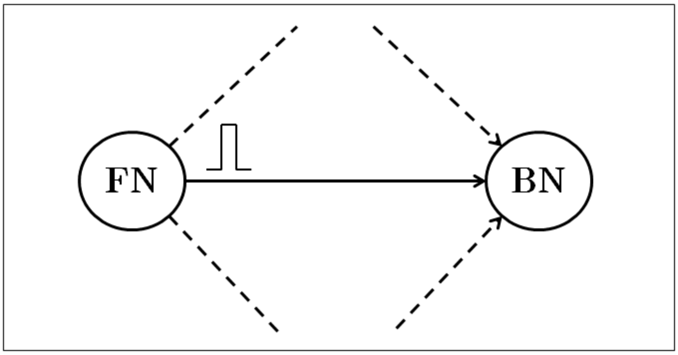


Fig. 1. A Firing Neuron (FN) emitting a pulse to a Burning Neuron (BN). The dashed lines indicate other connections linking the FN to other BNs or incoming connections from other FNs to the depicted BN. Of course, each neuron can be both FN and BN, depending on the direction of the activity.

The state of each burning neuron is evaluated through the following equations:

$$S = S_p + P_r P_w - T_l \ , \ \text{for } S < S_{th} \tag{1}$$

$$S = S_p + P_r P_w + T_r \ , \ \text{for } S \geq S_{th} \tag{2}$$

In (1)–(2), S denotes the inner state of the neuron; when $S = 0$ the neuron is in its resting state. S_p is the previous state, whereas, S_{th} represents the spiking threshold. The latter is conventionally chosen equal to $1 + d$, where d indicates a threshold constant. The quantity P_r, "presynaptic weight", represents the signal emitted by a firing neuron to a number of burning neurons; this quantity is conventionally equal to the inverse of the fan-out of the firing neuron, but other choices can be taken into account. Of course, this is a simplification by which we only consider inputs with the same amplitude. For the purpose of considering the inhibitory effect, P_r is chosen negative for inhibitory neurons. The quantity P_w, "postsynaptic weight", represents the connection strength between a couple of neurons. If P_w is equal to 0, the related connection is not present.

Finally, with T_l (*leakage term*) we take into account the subthreshold decay for the passive mode ($S < S_{th}$). In particular, we have chosen a linear decay behavior (this kind of decay is used in [17]), then $T_l = Ld\varDelta t$; in which Ld is the linear parameter, whereas, Δt represents the temporal distance between a couple of consecutive input spikes.

In active mode ($S \geq S_{th}$), the neuron is ready to fire: its firing is not instantaneous, but occurs after a continuous-time delay called *time-to-fire*. This quantity can be affected by inputs, making the neuron sensitive to possible changes in the network for a time window bounded by the time-to-fire itself. The inner state and the time-to-fire are related through the following bijective relationship, called *firing equation*.

$$t_f = \frac{1}{(S-1)} \; . \tag{3}$$

Equation (3) represent an approximation of the curve that we have obtained through the simulation of a membrane patch stimulated by brief current pulses (0.01 ms of duration), solving the Hodgkin-Huxley equations [5] making use of NEURON simulator [18]. Similar behaviors have been investigated by other authors; for example, in Wang et al. [19], using DC inputs (see Fig. 1 in [19]).

In Fig. 2, a qualitatively comparison between the simulated behavior of the latency and the firing equation is shown.

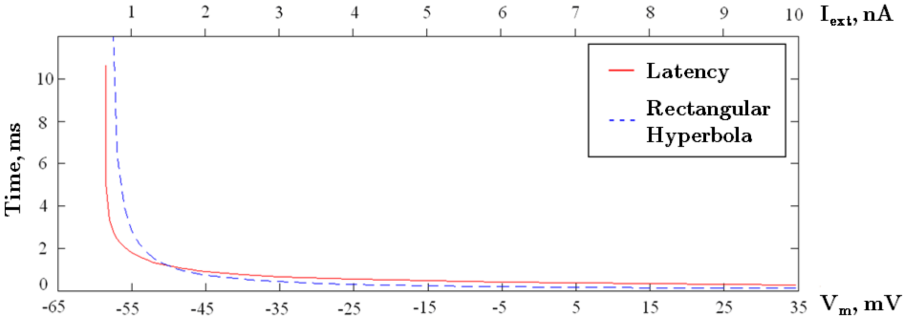


Fig. 2. The red line indicates the latency as a function of the membrane potential V_m (red marked scale), or else of the current amplitude I_{ext}, equivalently, obtained by means of simulations in NEURON environment. The dashed blue line indicates the rectangular hyperbola (i.e., the firing equation properly shifted for the comparison). In addition, the normalized scale S is reported (blue marked scale). Note that, below the S_{th} value no spike can be generated (fading blue area).

Note that, using (3) under proper considerations, it is possible to obtain T_r (*rise term*) in active mode, as follows:

$$T_r = \frac{(S_p-1)^2 \Delta t}{1-(S_p-1)\Delta t} \; . \tag{4}$$

in which S_p represents the previous state, whereas $\Delta t = t_c - t_p$ is the temporal distance between two consecutive incoming spikes; where t_c and t_p represent the times related to current and previous states, respectively.

Note that, the denominator of (4) must be positive, thus $\Delta t < 1/(S_p - 1)$ (i.e., $\Delta t < t_{fp}$, in which t_{fp} is the previous time-to-fire value).

Equation (4) allows us to determinate the inner state of a burning neuron at the moment when it receives further inputs during the t_f time window.

Finally, for $S = S_{th} = 1 + d$, the time-to-fire is equal to $t_{f,max} = 1/d$, representing the upper bound of the time-to-fire. The latter consideration is crucial in order to have a finite maximum latency [15].

In order to make more clear the behavior of the model, we have depicted the quantities introduced by means of (1)–(2)–(3) in Fig. 3. The effect of (4) is shown in Figs. 4a-b.

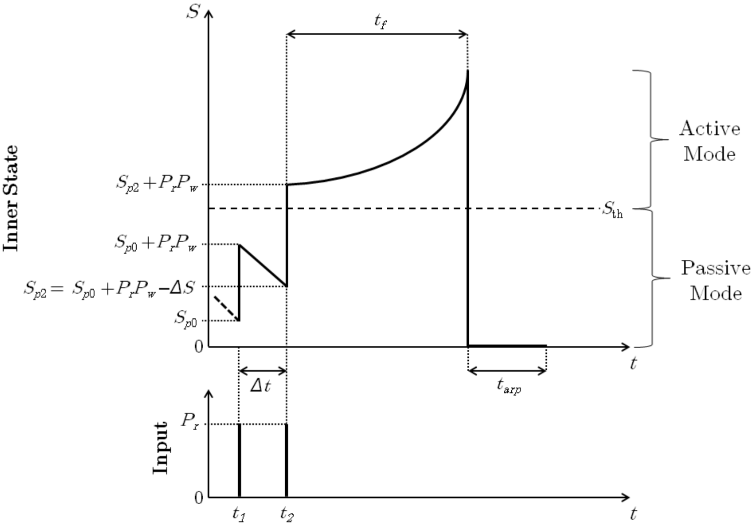


Fig. 3. In this figure, an example of the qualitatively inner state behavior of a neuron in passive and active modes is illustrated. An incoming excitatory input at t_1 causes an instantaneous increase of the state from S_{p0} to $S_{p0} + P_rP_{w1}$. At t_2 a second excitatory input is applied, then the state increase his value from S_{p2} to $S_{p2} + P_rP_{w2}$ (in this example, we have chosen $P_{w2} \neq P_{w1}$). Note that, $S_{p2} < (S_{p0} + P_rP_{w1})$; indeed, under the spiking threshold (S_{th}) the neuron is affected by a linear decay. Moreover, due to the latency effect, the firing is not instantaneous but occurs after t_f. Finally, after the firing, the neuron is reset to its resting potential (i.e., $S = 0$) for a time equal to t_{arp} (i.e., absolute refractory period).

Note that, excitatory (inhibitory) inputs increase (decrease) the inner state of a burning neuron. Therefore, when this neuron is in active mode, excitatory (inhibitory) inputs decrease (increase) the related time-to-fire (Fig. 4a and 4b, respectively).

Moreover, if the inhibitory effect is so strong to pull the burning neuron state under the spiking threshold, the time-to-fire will be canceled and the state will come back to the passive mode.

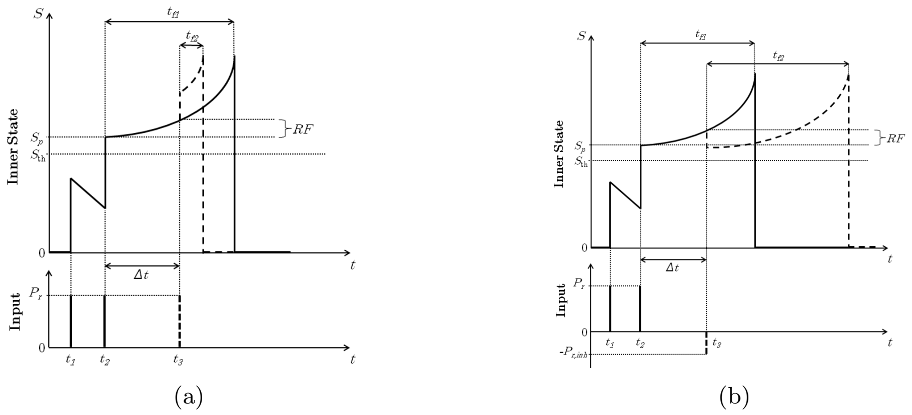


Fig. 4. (a) A hypothetical third excitatory input at t_3 (dashed pulse) would cause a reduction of the spike latency; then, the neuron would fire at $t = t_3 + t_{f2}$ (dashed line). (b) A hypothetical inhibitory input at t_3 (dashed pulse) would cause an increase of the spike latency $t = t_3 + t_{f2}$ (dashed line). In general, the amplitude of the presynaptic inhibitory input ($P_{r,inh}$) is different from the excitatory ones. In this case, a smaller value of $P_{r,inh}$ has been chosen. However, since in (a) and (b) we have assumed that each presynaptic input came from various synapses (implying different P_w values), then the $P_r P_w$ product is different. Also, T_r effect is shown.

3 Network Topology and Plasticity Rules

3.1 CNN-*like* Topology

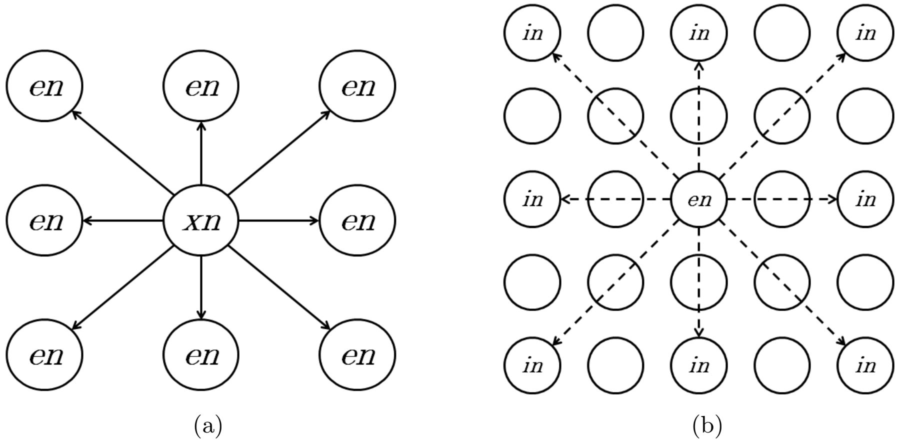
In order to show behaviors emerging from simulations conducted through the paradigm here proposed, in this section we present a model characterized by a CNN-*like* architecture topology. Therefore, a firing neuron emits its spikes to a number of burning neurons belonging to a neighborhood, "v". It is possible to change the size of the neighborhood by setting the parameter "v".

In Fig. 5 the grids for both excitatory (Fig. 5a) and inhibitory (Fig. 5b) neurons are illustrated.

Note that, in order to maintain the balance between excitation and inhibition, the number of inhibitory neurons is less than the excitatory ones, about 15%-25% of the global neuron population [20]. Typically, this represents a condition necessary but not sufficient to guarantee the network stability, as pointed out recently [21]. Synaptic plasticity provides a further contribution in order to maintain stable the network activity.

In the model, we have considered the following synapse classes: *excitatory-to-excitatory* (s_{ee}), *excitatory-to-inhibitory* (s_{ei}), and *inhibitory-to-excitatory* (s_{ie}). Note that, self-connections are not contemplated.

Since for simplicity we assumed that also inhibitory neurons are integrators, *inhibitory-to-inhibitory* synapses are not implemented. In fact, if this class were present, this would entail a reduction of the number of inhibitory neurons dur-



(a) (b)

Fig. 5. (a) Synapses grid for excitatory firing neurons (*en*); *xn* indicates an excitatory or inhibitory (*in*) firing neuron. (b) Synapses grid for inhibitory neurons.

ing the simulation, pushing the network activity toward instability due to an uncontrolled excitation.

In general, the spiking activity strictly depends on the synaptic circuitry [22], but in this work we have considered a regular topological structure rather than a biological one, which is too complex to reproduce. However, this choice does not affect the basic paradigm.

Finally, in further works we will simulate different topologies such as modular (e.g., [23]), hierarchical (e.g., [24]), small-world (e.g., [25,26]), etc.

3.2 Synaptic Plasticity Rules

For the purpose of taking into account the synaptic plasticity phenomenon, we propose the following simple rules. They represent a functional simplification of the *Postsynaptic Rule*: in its most general form, this rule states that it is the positional pattern and timing of heterosynaptic inputs with respect to the homosynaptic inputs to a given synapse that governs the change in postsynaptic efficacy induced by a modifying substance at that synapse [27].

- *Exponential decay.* All postsynaptic weights grow down to the minimum value in an exponential way with a proper time constant (τ).

$$P_w = P_{w,min} + (P_w - P_{w,min})\mathrm{e}^{-\frac{\Delta t}{\tau}} \ . \tag{5}$$

- *Homosynaptic enhancement.* When a spiking event occurs from a certain synapse, the postsynaptic weight grows up, in function of previous spikes on the same neuron and from the same synapse, occurred in a specified time window (homosynaptic window).

- *Heterosynaptic enhancement.* When a spiking event occurs from a certain synapse, the postsynaptic weight grows up, in function of previous spikes on the same neuron, in a specified time window (heterosynaptic window) from other synapses.

The growing rates related to homo- and heterosynaptic rules are properly applied using the following equation:

$$\varDelta P_w = \eta(P_{w,max} - P_w) \ . \tag{6}$$

with $0 < \eta \leq 1$, representing the *learning rate.* $P_{w,max}$ represents a cut-off value, but thanks to the exponential decay the saturation of the weights is avoided.

As show in subsection 4.2, these simple rules, together with the CNN-*like* topology, seem to be suitable to realize a *confinement-competition-selection model* [9].

In future works, we would also implement alternative strategies such as STDP (e.g., [28]) or Synaptic Scaling (e.g., [29]).

4 Event-Driven Simulations

4.1 Event-Driven Approach for the Network Simulation

As we have already stressed in the introduction, for the purpose of emulating a continuous-time behavior an event-driven approach is required [17], [30,31]. Therefore, a simple MATLAB code has been implemented, by means of which the simulation proceeds searching for the active neuron with the minimum time-to-fire, in order to determinate the next firing event to be scheduled in the spike timing array list. Then, the evaluation of firing event effects on all the directly burning neurons is made. A simulation procedure summary is illustrated below.

1. Pseudo-random inner state assignment for all neurons. Then, some neurons could be active when the simulation is running.
2. If all neuron states are less than the spiking threshold S_{th}, no active neuron is present. In this case, there is no activity.
3. The inputs are applied and the inner states S are computed for each burning neuron.
4. If a subset of neurons become over threshold, the time-to-fire for all active neurons will be evaluated, i.e. the following cyclic simulation procedure is applied:
   (a) Find the neuron N_0 with the minimum time-to-fire, t_{f0}. Apply to the global simulation time an increase equal to t_{f0}. According to this choice, update the time-to-fire and the inner states for each active neuron.
   (b) Firing of the neuron N_0. According to this event, make N_0 passive and update the states of all directly connected burning neurons. Update the postsynaptic weights according to the synaptic rules. Finally, the quantity t_{f0} is subtracted to the time-to-fire of all active neurons.
   (c) Update the set of the active neurons. If no active neuron is present the simulation is terminated. Otherwise, repeat the procedure from step (a).

4.2 Simulation Results

The whole distribution of the synapses is dynamically stored in a $N \times M$ matrix, called P_w (i.e., postsynaptic weights). Each entry of this matrix represents the weight of the particular synapse linking the firing neuron (j-th column of the matrix) to the burning one (i.e., i-th row of the matrix). Because of some synapses are not present (i.e., the network is not fully connected), some entries are equal to zero.

Defining *nen* as the number of excitatory neurons, *nin* as the number of inhibitory ones, *nef* as the number of external sources, and *nt* as the total number of neurons (including the external sources) the P_w matrix can be divided into submatrices:

$$P_w = \begin{pmatrix} P_{w_{11}} & P_{w_{12}} & P_{w_{13}} \\ P_{w_{21}} & P_{w_{22}} & P_{w_{23}} \end{pmatrix}$$

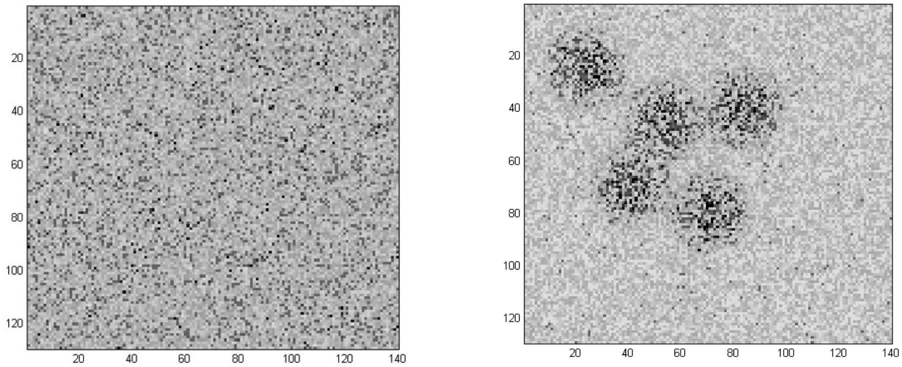
in which:

1. $P_{w_{11}}(pw_{i,j})$ represents the submatrix for the excitatory-to-excitatory synapses (s_{ee}), with $i = j = 1, ..., nen$; since no self-connection is considered, each entry of the main diagonal is equal to zero.
2. $P_{w_{12}}(pw_{i,j})$ represents the submatrix for the inhibitory-to-excitatory synapses (s_{ie}), with $i = 1, ..., nen$ and $j = nen + 1, ..., nen + nin$.
3. $P_{w_{13}}(pw_{i,j})$ represents the submatrix for the connections among external sources and excitatory neurons (s_{ese}), with $i = 1, ..., nen$ and $j = nen + nin + 1, ..., nt$.
4. $P_{w_{21}}(pw_{i,j})$ represents the submatrix for the excitatory-to-inhibitory synapses (s_{ei}), with $i = nen + 1, ..., nen + nin$ and $j = 1, ..., nen$.
5. $P_{w_{22}}(pw_{i,j})$ represents the submatrix for the inhibitory-to-inhibitory (s_{ii}), with $i = nen + 1, ..., nen + nin$ and $j = nen + 1, ..., nen + nin$, and it is a zero matrix as this class of connections is not considered here.
6. $P_{w_{23}}(pw_{i,j})$ represents the submatrix for the connections among external sources and inhibitory neurons (s_{esi}), with $i = nen + 1, ..., nen + nin$ and $j = nen + nin + 1, ..., nt$.

Note that, since the input signal cannot back-propagate, then the submatrices related to excitatory- and inhibitory-to-external sources, and external source self-connections are not present. Of course, *nen*, *nin* and *nef* can be chosen arbitrarily large.

Moreover, we have defined a $nt \times 5$ matrix, called S, in which are dynamically stored all the parameters related to all neurons of the network: the *state* (i.e., $S(:,1)$), the *time-to-fire* (i.e., $S(:,2)$), the *tlastfire* (i.e., $S(:,3)$, representing the time from the last spike emitted), the *tlastburning* (i.e., $S(:,4)$, representing the time from the last spike received) and the *presynaptic weight* P_r (i.e., $S(:,5)$), for each neuron. This list includes the external sources, but they are not affected by the same rules of the neurons (i.e., Eqs. (1)–(2)–(3)–(4)–(5)–(6)). Indeed, external sources are thought as access nodes by which we provide spike sequences to the network.

In the following figures, we show in a 2D map the spiking activity before and after stimulation. In this case, we have applied random spike sequences. Each point of the map represents the state level in gray scale: darker points imply a high activity. Note that, in order to avoid boundary effects, the 2D neuron map (obtained by the combination of the two grids shown in Figs. 5a–b) has been folded as a taurus.



(a)

(b)

Fig. 6. (a) Spontaneous spiking activity. (b) Formation of neuronal groups after stimulation.

For this preliminary study, the network was composed by 18060 excitatory and 2021 inhibitory neurons; moreover, we have applied 25 external sources in a pseudo-random fashion when the simulation was already run. The size of the 2D map was 140 × 129 (i.e., the size of the excitatory neuron grid). Finally, we have chosen a neighborhood $v = 4$, then each firing neuron could fire to 80 burning neurons (i.e., $[(2v + 1)^D - 1]$, with $D = 2$).

As regards the neuron model parameters, we have assumed a threshold constant $d = 0.04$, implying a spiking threshold $S_{th} = 1 + d = 1.04$, and a maximum time-to-fire $t_{f,max} = 1/d = 25$; a linear decay $Ld = 0.001$. In addition, we have considered the postsynaptic weights P_w in the range [0.1, 3].

In Fig. 6a a spontaneous activity of the network is depicted, which imply that some neurons were active when the simulation was run. We have also reported the spiking activity as a consequence of a stimulation consisting of pseudo-random spike sequences (Fig. 6b). Notice the emerging of 5 neuronal groups, in which the activity is higher than the rest of the network. Moreover, when we removed the external input these groups maintained their activity stable, preserving the shapes depicted in Fig. 6b. We believe that this confinement/selection behavior is due to both the architecture topology and synaptic plasticity rules implemented.

Of course, by means of proper techniques, further studies on the "memory" implications are needed (e.g., using statistical tools).

5 Conclusions

In this paper, we have introduced a simple paradigm in order to realize a continuous-time spiking neural network simulator. Since the simulations are conducted on a digital PC, we have implemented an *ad-hoc* event-driven method based on an array list. In this array, spike times of active neurons are stored and the algorithm proceeds searching for the minimum spike time in the list. Thus, a scheduling of the events is performed. Such a method allows us to implement a "continuous-time" behavior and to reduce the computational cost as well. Note that, event-driven simulations seem to be more suitable for the purpose of emulating the realistic dynamics of biological systems; indeed, as pointed out in the introduction, clock-driven simulations cause some relevant errors in the computation, which can mask the real network behavior.

We have taken into account some important neuronal characteristics such as subthreshold decay, spike latency, synaptic integration, excitatory and inhibitory effects, and synaptic plasticity. Even though we addressed the network implementation more from a functional point of view rather than a biological one, some interesting preliminary results have been obtained. In particular, formation and maintenance of neuronal groups after stimulation have been observed.

Further works will be focused on the statistical implications about the activity of these neuronal groups and the chance of storing information, realizing then *analog memories.*

References

1. Maass, W.: Networks of spiking neurons: The third generation of neural network models. Neural Netw. 10(9), 1659–1671 (1997)
2. Belatreche, A., Maguire, L.P., McGinnity, M.: Advances in design and application of spiking neural networks. Soft Computing - A Fusion of Foundations, Methodologies and Applications 11(3), 239–248 (2006)
3. Ponulak, F., Kasiński, A.: Introduction to spiking neural networks: Information processing, learning and applications. Acta Neurobiol. Exp. 71(4), 409–433 (2011)
4. Brunel, N., van Rossum, M.C.W.: Lapicque's 1907 paper: from frogs to integrate-and-fire. Biol. Cybern. 97(5-6), 337–339 (2007)
5. Hodgkin, A.L., Huxley, A.F.: A quantitative description of membrane current and application to conduction and excitation in nerve. J. Physiol. 117(4), 500–544 (1952)
6. Izhikevich, E.M.: Which Model to Use for Cortical Spiking Neurons? IEEE Trans. on Neural Networks 15(5), 1063–1070 (2004)
7. Izhikevich, E.M.: Polychronization: Computation with spikes. Neural Comput. 18(2), 245–282 (2006)
8. Chrol-Cannon, J., Gruning, A., Yaochu, J.: The emergence of polychronous groups under varying input patterns, plasticity rules and network connectivities. In: The 2012 International Joint Conference on Neural Networks (IJCNN), pp. 1–6. IEEE (2012)
9. Edelman, G.M.: Neural Darwinism: The Theory of Neuronal Group Selection. Basic Book, Inc., New York (1987)
10. Izhikevich, E.M., Gally, J.A., Edelman, G.M.: Spike-timing Dynamics of Neuronal Groups. Cerebral Cortex 14(8), 933–944 (2004)

11. Burkitt, A.N.: A review of the integrate-and-fire neuron model: I. Homogeneous synaptic input. Biol. Cybern. 95(1), 1–19 (2006)
12. Burkitt, A.N.: A review of the integrate-and-fire neuron model: II. Inhomogeneous synaptic input and network properties. Biol. Cybern. 95(2), 97–112 (2006)
13. Brette, R., Rudolph, M., Carnevale, T., Hines, H., Beeman, D., Bower, J.M., Diesmann, M., Morrison, A., Goodman, P.H., Harris Jr., F.C., Zirpe, M., Natschläger, T., Pecevski, D., Ermentrout, B., Djurfeldt, M., Lansner, A., Rochel, O., Vieville, T., Muller, E., Davison, A.P., El Boustani, S., Destexhe, A.: Simulation of networks of spiking neurons: A review of tools and strategies. J. Comput. Neurosci. 23(3), 349–398 (2007)
14. Citri, A., Malenka, R.C.: Synaptic plasticity: multiple forms, functions, and mechanisms. Neuropsychopharmacology 33(1), 18–41 (2008)
15. FitzHugh, R.: Mathematical models of threshold phenomena in the nerve membrane. Bull. Math. Biophys. 17(4), 257–278 (1955)
16. Chua, L., Yang, L.: Cellular Neural Networks: Theory. IEEE Trans. on Circuits and Systems 35(10), 1257–1272 (1988)
17. Mattia, M., Del Giudice, P.: Efficient event-driven simulation of large networks of spiking neurons and dynamical synapses. Neural Comput. 12(10), 2305–2329 (2000)
18. NEURON simulator, http://www.neuron.yale.edu/neuron/
19. Wang, H., Chen, Y., Chen, Y.: First-spike latency in Hodgkin's three classes of neurons. J. of Theoretical Biology 328, 19–25 (2013)
20. Okun, M., Lampl, I.: Balance of excitation and inhibition. Scholarpedia 4(8), 7467 (2009), http://www.scholarpedia.org/article/Balance_of_excitation_and_inhibition
21. Pernice, V., Staude, B., Cardanobile, S., Rotter, S.: Recurrent interactions in spiking networks with arbitrary topology. Physical Review E 85, 031916 (2012)
22. Buzsáki, G.: Rhythem of the brain. Oxford University Press, Inc. 198 Madison Avenue, New York (2006)
23. Parasuraman, K., Elshorbagy, A., Carey, S.: Spiking modular neural networks: a neural network modeling approach for hydrological processes. Water Resources Research 42(5), 1–14 (2006)
24. Wu, Q.X., McGinnity, M., Maguire, L., Cai, R., Chen, M.: Simulation of Visual Attention Using Hierarchical Spiking Neural Networks. In: Huang, D.-S., Gan, Y., Premaratne, P., Han, K. (eds.) ICIC 2011. LNCS, vol. 6840, pp. 26–31. Springer, Heidelberg (2012)
25. Watts, D.J., Strogatz, S.H.: Collective dynamics of "small-world" networks. Nature 393(1), 440–442 (1998)
26. Newman, M.E.J.: The structure and function of complex networks. SIAM Review 45(2), 167–256 (2003)
27. Finkel, L.H., Edelman, G.M.: Interaction of synaptic modification rules within populations of neurons. Proc. Natl. Acad. Sci. USA, 1291–1295 (1985)
28. Song, S., Miller, K.D., Abbott, L.F.: Competitive Hebbian learning through spike-timing-dependent synaptic plasticity. Nature 3(9), 919–926 (2000)
29. Sullivan, T.J., de Sa, V.R.: Homeostatic synaptic scaling in self-organizing maps. Neural Networks 19, 734–743 (2006)
30. Ros, E., Carrillo, R., Ortigosa, E.M., Barbour, B., Agís, R.: Event-Driven Simulation Scheme for Spiking Neural Networks Using Lookup tables to Characterize Neuronal Dynamics. Neural Comput 18(12), 2959–2993 (2006)
31. D'Haene, M., Schrauwen, B., Van Campenhout, J., Stroobandt, D.: Accelerating Event-Driven Simulation of Spiking Neurons with Multiple Synaptic Time Constants. Neural Comput. 21(4), 1068–1099 (2009)