

# Automata Computation with Hodgkin-Huxley Based Neural Networks Composed of Synfire Rings

Jérémie Cabessa

Laboratory of Mathematical Economics  
Université Paris 2 – Panthéon-Assas  
75006 Paris, France  
E-mail: jeremie.cabessa@u-paris2.fr

Aubin Tchaptchet

Institute of Physiology  
Philipps University of Marburg  
35037 Marburg, Germany  
E-mail: tchaptch@students.uni-marburg.de

**Abstract**—Recent results have shown that finite state automata can be simulated by recurrent neural networks composed of synfire rings. The simulation process was shown to work correctly in the cases of Boolean neural networks and of Izhikevich spiking neural networks. In this paper, we generalize these results to the very biological context of the Hodgkin-Huxley neural model. We prove that any finite state automaton can be simulated by a Hodgkin-Huxley based recurrent neural network composed of synfire rings. In this framework, the inhibitory system ensuring the transition between the successive rings can be significantly simplified. These results show that a neuro-inspired paradigm of abstract computation based on sustained activities of neural assemblies is indeed possible, and potentially harnessable. They also constitute a first step towards the implementation of biological neural computers.

## I. INTRODUCTION

It has early been observed that Boolean recurrent neural networks are computationally equivalent to finite state automata [1]–[3]: on the one hand, any Boolean recurrent neural networks can be simulated by some finite state automaton; on the other hand, any finite state automaton can be simulated by some Boolean network. The latter result opened the way to important further investigations, motivated by the possibility to implement finite state machines on parallel hardwares (see for instance [4]–[19]). Nowadays, the computational capabilities of diverse neural models have been shown to range from the finite automaton degree [1]–[3], [19], up to the Turing [20]–[26] or even to the super-Turing level [27]–[31]. These studies have been generalized to alternative bio-inspired paradigms of computation [31]–[37].

But the neural networks involved in the above mentioned studies are still far from the neurobiological reality. First of all, the discrete-time first-order neural model is highly simplistic. Moreover, in biological neural nets, information is more likely processed by cell assemblies rather than by isolated entities. Also, synaptic connections are unreliable. And neural nets are subjected to various phenomena, like synaptic plasticity. In a more biologically oriented context, the implementations of associative memory tasks, of logical gates, or of abstract devices have been achieved on diverse kinds of networks of oscillators [38]–[41]. Logical gates have also been physically implemented in patterned neural cultures [42], [43].

In terms of information processing, the concept of a *synfire chain* – a sequence of layers of neurons that are fully connected from one stratum to the next – has been proposed as a fundamental structure of biological neural nets. Synfire chains are indeed capable of conveying repeated complex spatio-temporal patterns of discharges in a robust and highly temporally precise way [44]–[48]. Besides, the spontaneous emergence of an abundance of “looping” synfire chains – referred to as *synfire rings* – has been observed in self-organizing neural networks subjected to various mechanisms of plasticity [49].

Based on these considerations, recent results have shown that finite state automata can be simulated by recurrent neural networks composed of synfire rings [50], [51]. The simulation process was shown to work correctly in the cases of Boolean neural networks [50] and of Izhikevich spiking neural nets [51]. This paradigm of neural computation finds its relevance at many levels: (i) The successive computational states are achieved via temporally robust activities of cell assemblies – the synfire rings – rather than by discrete spiking configurations. (ii) The successive computational states are encoded into cyclic attractor dynamics induced by the sustained activities of the synfire rings. (iii) The transitions between such attractors are perfectly controlled, in an input-driven way. (iv) The global computational process remains robust to various kinds of architectural failures and synaptic noises.

In this paper, we generalize these results to the very biological context of the Hodgkin-Huxley neural model. More precisely, we prove that any finite state automaton can be simulated by a Hodgkin-Huxley based recurrent neural network composed of synfire rings. In this framework, the inhibitory system ensuring the transition between the rings can be significantly simplified (compared with previous works [50], [51]), due to the consideration of the refractory period of the cells.<sup>1</sup> These results show that a neuro-inspired paradigm of abstract computation based on sustained activities of neural assemblies is indeed possible, and potentially harnessable. They also constitute a first step towards the implementation of biological neural computers.

<sup>1</sup>In fact, the so-called “triangular structures” of [50], [51], which were precisely the Achille’s heel of the construction’s robustness, are no more needed.

## II. FINITE STATE AUTOMATA AND BOOLEAN RECURRENT NEURAL NETWORKS

Boolean recurrent neural networks (BRNNs) are recurrent neural networks composed of McCulloch and Pitts cells [1]. At each time steps, the activation values of the cells is either firing (1) or quiet (0). Formally, the dynamics of the network is computed as follows: given the activation values of the input neurons  $(u_j(t))_{j=1}^M$  and the internal neurons  $(x_j(t))_{j=1}^N$  at time  $t$ , the activation values of the internal neurons  $(x_i(t+1))_{i=1}^N$  at time  $t+1$  are given by the following equations:

$$x_i(t+1) = \theta \left( \sum_{j=1}^N a_{ij} \cdot x_j(t) + \sum_{j=1}^M b_{ij} \cdot u_j(t) + c_i \right), \quad \text{for } i = 1, \dots, N \quad (1)$$

where the  $a_{ij}$ ,  $b_{ij}$ , and  $c_i$  are the synaptic weights and bias of the network, and  $\theta$  is the classical hard-threshold activation function defined by

$$\theta(x) = \begin{cases} 0 & \text{if } x < 1 \\ 1 & \text{if } x \geq 1. \end{cases}$$

Boolean recurrent neural networks are known to be computationally equivalent to finite state automata [1]–[3].

**Theorem 1** (Minsky 1967). *Any Boolean neural network can be simulated by some finite state automaton, and any finite state automaton can be simulated by some Boolean network.*

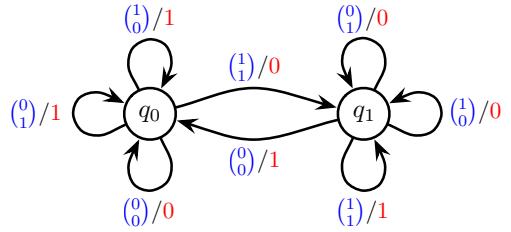
The first part of this statement is straightforward. A Boolean network with  $N$  cells has at most  $2^N$  spiking configurations. It can therefore be simulated by a finite state automaton with less than  $2^N$  states. The second part of the statement is more relevant, since it concerns the issue of the implementation of finite state machines on parallel hardwares (see for instance [4]–[19]). This possibility to simulate finite automata by recurrent neural networks constitutes the core of this work.

In order to illustrate this process, we consider a *transducer*, i.e., a finite state automaton provided with an output channel, implementing a serial binary adder. The transducer is represented as a labelled directed graph illustrated in Figure 1. The nodes and edges of the graph represent the computational states and transitions of the transducer. This transducer is composed of two states corresponding to the two situations of either being currently carrying a 1 in the adding process or not. A transition from node  $q_i$  to node  $q_j$  labelled by  $i/o$  means that if the transducer is in state  $q_i$  and it receives input symbol  $i$ , then it will output symbol  $o$  and move to state  $q_j$ .

The computation of the following binary sum  $s$

$$\begin{array}{r} 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 0 \ 1 \ 1 \\ + \ 1 \ 1 \ 1 \ 0 \ 0 \ 1 \ 1 \ 0 \ 0 \\ \hline 1 \ 1 \ 0 \ 0 \ 1 \ 1 \ 1 \ 1 \ 0 \end{array}$$

by the transducer of Figure 1 is illustrated in Table 1 (three first rows). The transducer starts in its initial state  $q_0$ . It receives as successive inputs the successive pairs of bits of  $s$  in reverse order, namely  $(1), (0), (1), (1), (0), (1), (1)$ , and receives a last



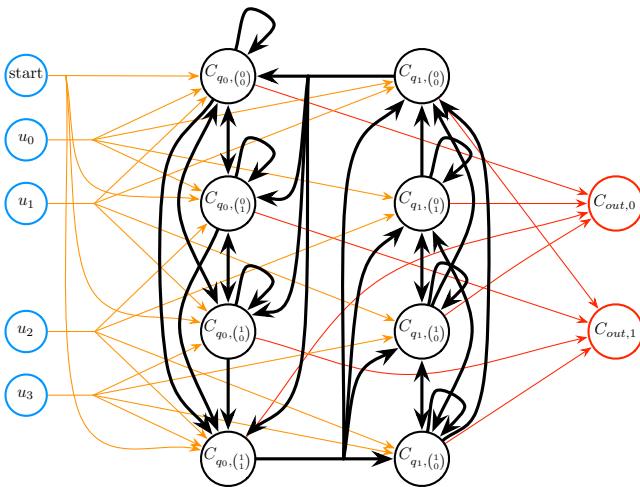
**Fig. 1:** A finite transducer implementing a serial binary adder. The nodes and edges of the graph represent the computational states and transitions of the transducer, respectively. A transition from  $q_i$  to  $q_j$  labelled by  $i/o$  means that if the transducer is in state  $q_i$  and it receives input symbol  $i$ , then it will output symbol  $o$  and move to state  $q_j$ . The initial state is  $q_0$ . This transducer can compute the sum of any two binary numbers. For this purpose, it starts from initial state  $q_0$  and takes as successive inputs the successive pairs of bits of the sum in reverse order; if needed, it adds a last input  $(0)$  so as to come back to state  $q_0$ . The successive output bits correspond to the result of the sum in reverse order.

input  $(0)$  in order to come back to the final state  $q_0$ . During this process, it moves from one state to the other according to its labelled transitions, and outputs the successive bits 0, 1, 1, 0, 0, 1, 1. This sequence of bits corresponds to the result of the sum  $s$  in reverse order.

A Boolean recurrent neural network simulating the transducer of Figure 1 is illustrated in Figure 2. This network is obtained on the basis on Minsky's original construction [3] which is not optimal in terms of number of cells and connections. The network has 4 input cells (blue) and two output cells (red) used to encode the 4 possible inputs and 2 possible outputs of the transducers, respectively. It also has 8 internal cells  $C_{s,i}$  (black) used to represent the 8 possible events of the transducer, i.e., the events “being in state  $s$  and receiving input  $i$ ”, for all  $s \in \{q_0, q_1\}$  and  $i \in \{(0), (1), (0), (1)\}$ . The network is designed in such a way that, at each time step, at most one internal cell and one output cell is spiking. More precisely, if at time step  $t$ , the network has its cell  $C_{s,i}$  being spiking and if it receives the encoding of input  $i'$ , then at time  $t+1$ , cells  $C_{s',i'}$  and  $C_{out,o}$  will be spiking, where  $s'$  and  $o$  are given by the transducer's transition  $(s, i/o, s')$ .

The computation of the binary sum  $s$  (given above) by the Boolean network of Figure 2 is illustrated in Table 1 (eight middle rows). The network starts with all cells being quiet. It receives the successive input patterns  $(1, 0, 0, 1, 1)^T$ ,  $(0, 1, 1, 0, 0)^T$ ,  $(0, 0, 1, 0, 1)^T$ ,  $(0, 0, 0, 1, 1)^T$ ,  $(0, 1, 0, 1, 0)^T$ ,  $(0, 0, 0, 1, 1)^T$ ,  $(0, 1, 1, 0, 0)^T$  which correspond to the encodings the successive pairs of bits of  $s$  in reverse order  $(1), (0), (1), (1), (0), (1), (1)$ . Its dynamics is then governed by Equation (1). The (non-quiet) output patterns that it produces  $(1), (0), (1), (0), (1), (0), (1), (0)$  correspond to the encodings of the successive bits 0, 1, 1, 0, 0, 1, 1. This sequence of bits is the result of the sum  $s$  in reverse order.

These considerations show that the transducer of Figure 1 is perfectly simulated by the Boolean neural network of Figure 2 with a time delay of 1–2 time steps. More precisely, during



**Fig. 2:** A Boolean recurrent neural network computationally equivalent to the finite transducer of Figure 1. The four input cells (blue) are used to encode the four possible inputs of the transducer: the patterns  $(u_0 = 1, u_1 = 1, u_2 = 0, u_3 = 0)$ ,  $(u_0 = 1, u_1 = 0, u_2 = 1, u_3 = 0)$ ,  $(u_0 = 0, u_1 = 1, u_2 = 0, u_3 = 1)$  and  $(u_0 = 0, u_1 = 0, u_2 = 1, u_3 = 1)$  encode the transducer's inputs  $(\frac{0}{0})$ ,  $(\frac{0}{1})$ , and  $(\frac{1}{0})$  and  $(\frac{1}{1})$ , respectively. The “start” cell spikes only at time  $t = 0$  in order to initiate the dynamics. The internal cells (black) are organized in a  $4 \times 2$  grid-like structure which refers to the 4 possible inputs and 2 possible states of the transducer (Minsky's construction [3]). Cell  $C_{s,i}$  represents the event of the transducer being in state  $s$  and receiving input  $i$ , for all  $s \in \{q_0, q_1\}$  and  $i \in \{\frac{0}{0}, \frac{0}{1}, \frac{1}{0}, \frac{1}{1}\}$ . The two output cells (red) are used to encode the outputs of the transducer: the patterns  $(R_{out,0} = 1, R_{out,1} = 0)$  and  $(R_{out,0} = 0, R_{out,1} = 1)$  encode the transducer's outputs 0 and 1, respectively. The orange and black synaptic connections have weights  $1/3$ . The red ones have weights 1.

the computation, the transducer is in state  $s$ , receiving input  $i$  and producing output  $o$  at time  $t$  if and only if the neural network has its cells  $C_{s,i}$  and  $C_{out,o}$  spiking at times  $t + 1$  and  $t + 2$ , respectively. This feature can be verified in Table 1.

Finally, note the above construction is generic and can be applied to any finite state automaton. The formal proof of the correctness of this general simulation process goes back to Minsky's work [3].

### III. FINITE STATE AUTOMATA AND HODGKIN-HUXLEY NEURAL NETWORKS COMPOSED OF SYNFIRES RINGS

The equivalence between automata and recurrent neural networks presented in Section II has been extended to the more biological context of recurrent neural networks composed of synfire rings [50], [51]. Here, we show that these latter results can be further generalized to the even more biological context of the Hodgkin-Huxley (HH) model. More precisely, we show that any finite state automaton can be simulated by a recurrent neural network composed of synfire rings, and where the cells' dynamics are governed by the HH-equations. Formally, the following result holds.

**Theorem 2.** Any finite state automaton can be simulated by a Hodgkin-Huxley based recurrent neural network composed of synfire rings.

The rest of this section is devoted to the proof of this theorem. It presents the key steps of a construction process that starts from a given finite state automaton and builds a corresponding HH-based neural network composed of synfire rings which simulates this automaton correctly. The construction process is generic and can be applied to any finite state automaton. Section IV illustrates the correctness of this construction process when applied to the specific example of the serial binary adder of Figure 1.

#### A. Hodgkin-Huxley cells

The pioneering Hodgkin-Huxley<sup>2</sup> (HH) model is considered amongst the most accurate model for the simulation of biological neurons [52]. The parameters of the original conductance-based HH equations are highly precise for the modelling of action potentials. But they remain difficult to be adjusted when approaching and simulating some experimental data.

Based on physiological considerations, we chose to use a slightly simplified HH-model described in details in [53]. This simplified model replaces the rate constants by Boltzmann functions and discards the power functions, which makes it easier to handle, especially in the context of experimental data. The neurons of our HH-based neural network are all identical and the values of their parameters are those of the “standard neuron” of the virtual laboratory “SimNeuron” ([www.virtual-physiology.com](http://www.virtual-physiology.com)).

#### B. Synfire rings

A *synfire chain* consists of a sequence of layers of neurons that are fully connected from one stratum to the next by means of excitatory synaptic connections [44]–[46]. A *synfire ring* is a synfire chain that loops back in on itself, i.e., where the last layer is connected to the first [49]. In a synfire chain or ring, the weights of the connections are assumed to be strong enough to ensure that a spiking activity can propagate from one layer to the next in a robust manner. The internal connections of a synfire rings are referred to as the *intra-ring connections*. A synfire chain and a synfire ring are illustrated in Figure 3.

#### C. General construction

We describe the construction process to obtain a HH-based neural network composed of synfire ring capable of simulating any given finite state automaton. The main idea of the construction is to replace each cell  $C_{s,i}$  and  $C_{out,o}$  of Minsky's construction (illustrated in Figure 2) by a corresponding synfire ring  $R_{s,i}$  and  $R_{out,o}$ , respectively. The input cells of Minsky's construction remain however unchanged. In addition, every synaptic connection is replaced by a fibre of excitatory connection, and in some cases also, by an additional reverse

<sup>2</sup>Alan Llyod Hodgkin and Andrew Fielding Huxley were awarded of the Medicine's Nobel Prize in 1963 for this model.

**Table 1:** Computation of the sum  $s + \begin{smallmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \end{smallmatrix} = 1100110$  by the transducer of Figure 1, by the Boolean neural network of Figure 2 and by the HH-neural network of Figure 7 (see Sections III and IV for this last case). The three first rows illustrate the computation of the transducer. The next eight rows illustrate the computation of the Boolean neural network of Figure 2. We see that the transducer is correctly simulated by the Boolean neural network with a short delay of 1 or 2 time steps. The last two rows illustrate the computation of the HH-based neural network of Figure 7 (relative to Section III and IV). The time steps are no more relevant in this case.

time steps	0	1	2	3	4	5	6	7	8	9
states	$q_0$	$q_1$	$q_0$	$q_0$	$q_1$	$q_1$	$q_1$	$q_0$	–	–
inputs	$(1)$	$(0)$	$(1)$	$(1)$	$(0)$	$(1)$	$(0)$	–	–	–
outputs	0	1	1	0	0	1	1	–	–	–
◀ ..... result of the sum $s$ in reverse order ..... ▶										
cell <i>start</i>	1	0	0	0	0	0	0	0	0	0
cell $u_0$	0	1	0	0	1	0	1	0	0	0
cell $u_1$	0	1	1	0	0	0	1	0	0	0
cell $u_2$	1	0	0	1	1	1	0	0	0	0
cell $u_3$	1	0	1	1	0	1	0	0	0	0
cells $C_{s,i}$	–	$C_{q_0,(1)}$	$C_{q_1,(0)}$	$C_{q_0,(1)}$	$C_{q_0,(1)}$	$C_{q_1,(0)}$	$C_{q_1,(1)}$	$C_{q_1,(0)}$	–	–
cell $C_{out,0}$	0	0	1	0	0	1	1	0	0	0
cell $C_{out,1}$	0	0	0	1	1	0	0	1	1	0
◀ ..... encoding of the result of the sum $s$ in reverse order ..... ▶										
The next lines are relative to Sections III and IV. In this case, the time steps are no more relevant.										
rings $R_{s,i}$	–	$R_{q_0,(1)}$	$R_{q_1,(0)}$	$R_{q_0,(1)}$	$R_{q_0,(1)}$	$R_{q_1,(0)}$	$R_{q_1,(1)}$	$R_{q_1,(0)}$	–	–
ring $R_{out,j}$	–	$R_{out,0}$	$R_{out,1}$	$R_{out,1}$	$R_{out,0}$	$R_{out,0}$	$R_{out,1}$	$R_{out,1}$	–	–
◀ ..... encoding of the result of the sum $s$ in reverse order ..... ▶										

fibre of inhibitory connections, as described in details in the next paragraphs. The synfire rings will always be activated via fibres of excitatory connections projecting onto one specific of their layers, called the *activation layer* (dark blue filled cells in Figures 4, 5 and 6). We assume that the rings are wired in such a way that the information propagates inside them in the clockwise direction of rotation (grey or red little arrows in Figures 4, 5 and 6). According to this construction process, the HH-based neural network composed of synfire rings corresponding to the network of Figure 2 is schematically illustrated in Figure 7.

Each input connection of Minsky's construction between two cells  $u_i$  and  $C_j$  (orange arrows of Figure 2) is replaced by a fibre of excitatory connections projecting from the input cell  $u_i$  onto the activation layer of the targeted ring  $R_j$  (orange arrows of Figure 7). These *input connections* are illustrated in Figure 4. The synaptic weights are chosen such that one or even two fibres of input connections are not sufficient to activate the ring onto which they project.

Each internal connection of Minsky's construction between two cells  $C_i$  and  $C_j$  (black arrows of Figure 2) is replaced by a fibre of excitatory connections projecting from the activation layer of ring  $R_i$  onto that of ring  $R_j$ , and by a reverse fibre of inhibitory connections projecting from a layer of  $R_j$  located “after” its activation layer onto a layer of  $R_i$  located “before” its activation layer (these couple of fibres are represented by the black arrows of Figure 7), where “before” and “after” are relative to the clockwise direction of rotation. These *inter-ring connections* are illustrated in Figure 5. The excitatory synaptic weights are chosen such that one fibre of inter-ring connections

is not sufficient to activate the ring onto which it projects. The inhibitory synaptic weights are, by contrast, chosen such that if ring  $R_j$  becomes active via its activation layer, then it will send via one of its next layers a sufficiently strong inhibition to a layer of  $R_i$  located before its activation layer in order to kill the activation that this latter might receive a bit later. Note that this inhibition process is repeated as long as  $R_j$  remains active. In this way, the activation of a subsequent ring always triggers the inhibition of the previous one, ensuring that except during the transition phases at most one internal synfire ring is always active.

Each output connection of Minsky's construction between an internal cell  $C_i$  and an output cell  $C_j$  (red arrows of Figure 2) is replaced by a fibre of excitatory connections projecting from the activation layer of ring  $R_i$  onto that of ring  $R_j$  (red arrows of Figure 7). In this case, the excitatory synaptic weights are chosen such that one fibre of inter-ring connections is sufficient to activate the output ring onto which it projects. In addition, two fibres of inhibitory connections of the same kinds as that described above are introduced between all pairs of output rings. In this way, the activation of an output ring always triggers the inhibition of all other ones. These *output inter-ring connections* are illustrated in Figure 6.

The parameters of the HH-cells and the weights of the intra-ring connections are set such that any activated synfire ring will necessarily settle into a self-sustained activity, as long as it is not affected by any inhibitory inter-ring fibre of connections. The weights of the input and inter-ring excitatory connections are chosen such that the conjunction of two input fibres and one inter-ring fibre is sufficient to activate the ring onto which

it projects. By suitably tuning the parameters of the cells and connections, these conditions can always easily be fulfilled within our HH-based model.

According to this construction, any finite state automaton  $\mathcal{A}$  is simulated by its corresponding HH-neural network of synfire rings  $\mathcal{N}$  in the following precise sense. Suppose that  $\mathcal{A}$  receives the sequence of input symbols  $i = (i_0, i_1, \dots, i_n)$  inducing the computation

$$s_{in} \xrightarrow{i_0/o_0} s_{k_0} \xrightarrow{i_1/o_1} s_{k_1} \dots \xrightarrow{i_n/o_n} s_{k_n} \quad (2)$$

where each  $s_{k_m}$  and  $o_m$  is the successive state and output induced by input  $i_m$ , respectively. The way that  $\mathcal{N}$  is constructed ensures that if it receives the sequence of input patterns  $i' = (i'_0, \dots, i'_n)$ , where each  $i'_m$  is the encoding of  $i_m$ , then the sequences of internal and output synfire rings that will be activated are

$$R_{s_{in}, i_0}, R_{s_{k_0}, i_1}, \dots, R_{s_{k_{n-1}}, i_n} \quad (3)$$

$$R_{out, o_0}, R_{out, o_1}, \dots, R_{out, o_n} \quad (4)$$

In other words, the successive states and outputs of  $\mathcal{A}$  are perfectly reflected by the sequence of internal and output rings of  $\mathcal{N}$  that are activated: in fact, the states and outputs of  $\mathcal{A}$  correspond precisely to the indices of the internal and output rings of  $\mathcal{N}$  that are activated.

The simulation of the transducer of Figure 1 by the HH-based neural network of Figure 7 is illustrated in Table 1 (first three rows and last two rows). On the one hand, the computation of the binary sum  $s$  by the transducer corresponds to the following path (three first rows of Table 1)

$$q_0 \xrightarrow{(1)/0} q_1 \xrightarrow{(0)/1} q_0 \xrightarrow{(1)/1} q_0 \xrightarrow{(1)/0} q_1 \xrightarrow{(0)/0} q_1 \xrightarrow{(1)/1} q_1 \xrightarrow{(0)/1} q_0$$

On the other, the activity of the HH-based neural network computing the binary sum  $s$  is described as follow. The network starts with all cells being quiet. At sufficiently distant time steps – in order for it to have enough time to settle into the successive self-sustain activities induced by the synfire rings – the network receives the input patterns  $(1, 0, 0, 1, 1)^T$ ,  $(0, 1, 1, 0, 0)^T$ ,  $(0, 0, 1, 0, 1)^T$ ,  $(0, 0, 0, 1, 1)^T$ ,  $(0, 1, 0, 1, 0)^T$ ,  $(0, 0, 0, 1, 1)^T$ ,  $(0, 1, 1, 0, 0)^T$  which correspond to the encodings the successive pairs of bits of  $s$  in reverse order  $(\frac{1}{1}), (\frac{0}{0}), (\frac{1}{0}), (\frac{1}{1}), (\frac{0}{1}), (\frac{1}{1}), (\frac{0}{0})$ . According to the network's dynamics, the successive internal and output rings that are activated are the following (two last rows of Table 1)

$$R_{q_0, (\frac{1}{1})}, R_{q_1, (\frac{0}{0})}, R_{q_0, (\frac{1}{0})}, R_{q_0, (\frac{1}{1})}, R_{q_1, (\frac{0}{1})}, R_{q_1, (\frac{1}{1})}, R_{q_1, (\frac{0}{0})}$$

$$R_{out, 0}, R_{out, 1}, R_{out, 1}, R_{out, 0}, R_{out, 0}, R_{out, 1}, R_{out, 1}$$

Hence, the successive states and outputs of the transducer are perfectly reflected by the sequence of activated internal and output rings of the network, in the precise sense formalized in Relations (2) and (3-4) above. In addition, note that the successive activated output rings correspond to the encodings of the successive bits  $0, 1, 1, 0, 0, 1, 1$ . This sequence of bits is the result of the sum  $s$  in reverse order.

Due to space restrictions, the construction process that we described in this section cannot be formalized in further details, and its formal proof of correctness is postponed to a forthcoming journal paper. But we claim that the proposed construction is generic and can be applied to any finite state automaton. These considerations conclude the (still informal) proof of Theorem 2.

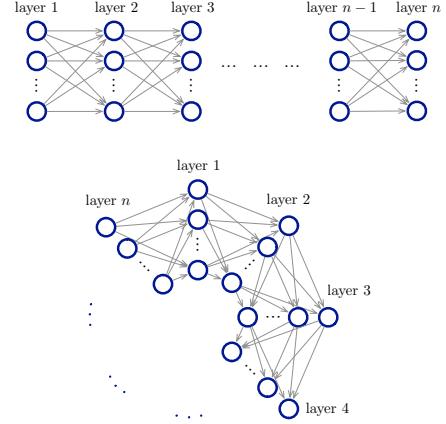


Fig. 3: A synfire chain and a synfire ring with  $n$  layers.

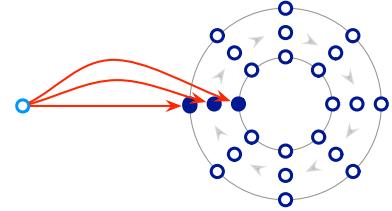


Fig. 4: A fibre of input excitatory connections (red arrows) projecting from an input cell  $u_i$  (light blue cell) onto the activation layer of a synfire ring  $R_i$  (dark blue filled cells).

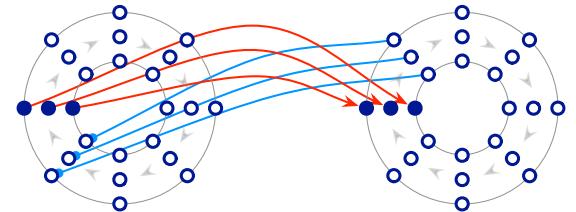
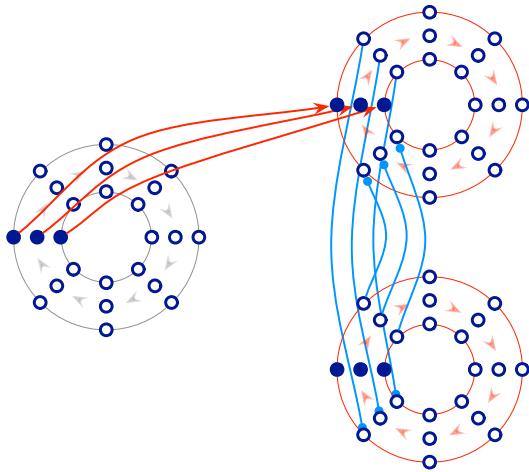


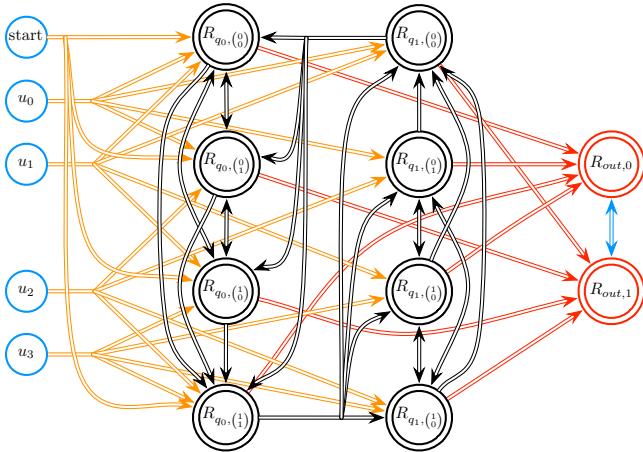
Fig. 5: Inter-ring excitatory and inhibitory connections between two synfire rings  $R_i$  and  $R_j$ . There is a fibre of excitatory connections (red arrows) projecting from the activation layer of  $R_i$  onto that of  $R_j$ . There is also a fibre of inhibitory connections (light blue arrows) projecting from a layer of  $R_j$  located “after” the activation layer onto a layer of  $R_i$  located “before” the activation layer.

#### IV. SIMULATION

In order to attest the correctness of our construction process, we show that the HH-based neural network of Figure 7



**Fig. 6:** Output inter-ring excitatory and inhibitory connections. The grey and red rings represent one internal and two output rings  $R_i$ ,  $R_{o1}$ ,  $R_{o2}$ , respectively. There is a fibre of excitatory connections (red arrows) projecting from the activation layer of  $R_i$  onto that of  $R_{o1}$ . There are also two fibres of inhibitory connections (light blue arrows) between  $R_{o1}$  and  $R_{o2}$ . Each one projects from a layer located “after” the activation layer onto a layer located “before” the activation layer.



**Fig. 7:** Schematic representation of the HH-neural network composed of synfire rings corresponding to the network of Figure 2. The blue circles are the input HH-cells. The black and red double circles represent the internal and output synfire rings made up of HH-cells, respectively. The blue, black and red arrows represent the fibres of input, inter-ring and output connections, described in Figures 4, 5 and 6, respectively. Note that for each inter-ring excitatory fibre projecting from a ring  $R_i$  to a ring  $R_j$  (black connection), there is a corresponding reverse inhibitory fibre from  $R_j$  to  $R_i$  that is not represented (cf. Figure 5). The blue fibres between  $R_{out,0}$  and  $R_{out,1}$  are the inhibitory output connections described in Figure 6.

simulates the automaton of Figure 1 in a adequate way, in the precise sense described at the end of Section III.

Our HH-network is subjected to 4 input signals (start signal is omitted) and is composed of 8 internal and 2 output synfire rings (cf. Figure 7). Each input signal is given in the form of a

constant external current stimulus with an amplitude of 1.9 nA and a duration of 1 ms. Note that one external current stimulus is not sufficient to generate an action potential of a single cell. Each ring consists in 12 layers of 3 cells each, which amounts to a total of 36 cells. The number of 12 layers has been chosen in such a way that the refractory period of the neurons could never inhibit the spike propagation throughout the layers. Accordingly, each activated synfire rings will necessarily settle into a self-sustained activity, as long as it does not receive any other inhibition.

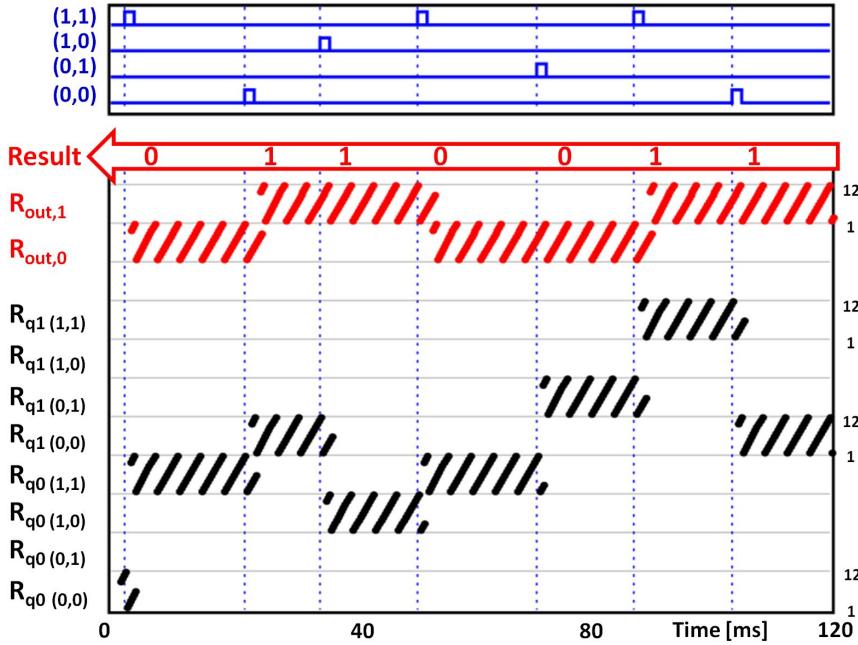
The strength of the synaptic connections are modelled by the alpha-function  $\alpha(t) = a \cdot t \cdot e^{-b \cdot t}$ , where  $a$  and  $b$  are different parameters depending on the kinds of connections that we consider. For the excitatory intra-ring connections (grey connections of Figure 3),  $a = 25.0$  nA and  $b = 2.0$  ms $^{-1}$  with a maximal amplitude  $I = 4.5$  nA. For the excitatory inter-ring connections (red connections of Figure 5),  $a = 3.0$  nA and  $b = 0.7$  ms $^{-1}$  with  $I = 1.7$  nA. And for the inhibitory inter-ring connections (blue connections of Figure 5),  $a = 15.0$  nA and  $b = 1.5$  ms $^{-1}$  with  $I = 3.5$  nA.

We simulated the activity of the network during the computation of the sum  $s$ . The raster plot of this simulation is presented in Figure 8. We submitted to the network a sequence of input patterns corresponding to the encodings the successive pairs of bits of  $s$  in reverse order. The sequence of internal and output rings that are activated corresponds precisely to that described in Table 1 (last two rows). In particular, the activity of the output rings (red dots) corresponds to the encoding of the correct result of  $s$  in reverse order. We simulated the network on other inputs and the result was always correct. This simulation shows that the automaton of Figure 1 is correctly simulated by our HH-based neural network composed of synfire rings.

## V. DISCUSSION

We showed that any finite state automaton can be simulated by a Hodgkin-Huxley based recurrent neural network composed of synfire rings. These results find their relevance at many levels. First, the classical equivalence between finite state automata and Boolean neural networks [3] (an its subsequent improvements [4]–[19]) would probably not so easily generalize to the context of more biological models, due to the simplicity of the first-order or second-order discrete-time neural models. By contrast, our work shows that finite state automata can be simulated by bio-inspired neural networks. Secondly, synfire chains and synfire rings are likely to be fundamental structures of biological neural networks, and a computational paradigm based on these structures possesses the following advantages:

- the successive computational states are achieved via temporally robust activities of cell assemblies;
- the successive computational states are encoded into cyclic attractor dynamics induced by the self-sustained activities of the rings;
- the transitions between such attractors are perfectly controlled, in an input-driven way;



**Fig. 8:** Raster plot of the activity of the HH-based neural network of Figure 7 simulating the automaton of Figure 1. The dashed blue lines represent the triggered input signal. Each input pattern  $(u_0 = 1, u_1 = 1, u_2 = 0, u_3 = 0)$ ,  $(u_0 = 1, u_1 = 0, u_2 = 1, u_3 = 0)$ ,  $(u_0 = 0, u_1 = 1, u_2 = 0, u_3 = 1)$  or  $(u_0 = 0, u_1 = 0, u_2 = 1, u_3 = 1)$ , which encode the automaton's inputs  $(^0_0)$ ,  $(^0_1)$ , and  $(^1_0)$  or  $(^1_1)$ , respectively, is represented as a single blue trigger input signal. Each dot in the lower diagram represents a spiking neuron of the network. The black dots represent the internal neurons and the red dots the output neurons. In this case, the input stream corresponds to the encoding of the binary sum  $s + 1111001$ : the successive pairs of bits of  $s$  are given in reverse order. We see that the sequence of activated output rings corresponds to the encoding of the result of  $s$  in reverse order, namely 1100110.

- the global computational process remains robust to various kinds of architectural failures and synaptic noises.

For future work, we envision to study the learning capabilities of the synfire ring neural architecture. In this case, one would need to consider a synfire ring architecture where the inter-ring connectivity is subjected to some form of Spike-Timing Dependent Plasticity (STDP). Then, it would be interesting to investigate if, starting from fully or randomly connected synfire based architectures, the networks could be capable of learning a relationship between certain input streams and synfire patterns of activity, in the form of an associative memory.

We also plan to generalize the present considerations towards the achievement of Turing-complete computation. For this purpose, the notion of “unboundedness” must be introduced somewhere, in order to model the infinite tapes of the Turing machines and their potentially unbounded contents. A first way would be to consider a synfire ring based architecture for which additional rings can appear throughout the computational process, in an unbounded way. A second way would be to use an additional information, like the synaptic strengths of certain specific connections for instance, that would be modelled as rational numbers and updated throughout the computational process.

With these achievements, we do not intend to argue that brain computational processes really proceed via simulations

of finite state automata in the very way that we described. Rather, our intention is to show that a neural paradigm of abstract computation based on sustained activities of cell assemblies is indeed possible, and potentially exploitable. As a consequence, biological neural networks should in principle be capable of simulating finite state automata, whether via the proposed paradigm, or via some other one. From a broader perspective, these results might lay the theoretical bases for the realization of biological neural computers.

#### ACKNOWLEDGMENT

This work was partially supported by the DARPA program Lifelong Learning Machines (L2M), BAA No. HR001117S0016 and cooperative agreement No. HR0011-18-2-0023.

#### REFERENCES

- [1] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity,” *Bulletin of Mathematical Biophysics*, vol. 5, pp. 115–133, 1943.
- [2] S. C. Kleene, “Representation of events in nerve nets and finite automata,” in *Automata Studies*, C. Shannon and J. McCarthy, Eds. Princeton, NJ: Princeton University Press, 1956, pp. 3–41.
- [3] M. L. Minsky, *Computation: finite and infinite machines*. Englewood Cliffs, N. J.: Prentice-Hall, Inc., 1967.
- [4] A. Cleeremans, D. Servan-Schreiber, and J. L. McClelland, “Finite state automata and simple recurrent networks,” *Neural Computation*, vol. 1, no. 3, pp. 372–381, 1989.
- [5] J. L. Elman, “Finding structure in time,” *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990.

- [6] N. Alon, A. K. Dewdney, and T. J. Ott, "Efficient simulation of finite automata by neural nets," *J. ACM*, vol. 38, no. 2, pp. 495–514, 1991.
- [7] J. B. Pollack, "The induction of dynamical recognizers," *Machine Learning*, vol. 7, pp. 227–252, 1991.
- [8] C. L. Giles, C. B. Miller, D. Chen, H. Chen, G. Sun, and Y. Lee, "Learning and extracting finite state automata with second-order recurrent neural networks," *Neural Computation*, vol. 4, no. 3, pp. 393–405, 1992.
- [9] R. L. Watrous and G. M. Kuhn, "Induction of finite-state languages using second-order recurrent networks," *Neural Computation*, vol. 4, no. 3, pp. 406–414, 1992.
- [10] Z. Zeng, R. M. Goodman, and P. Smyth, "Learning finite state machines with self-clustering recurrent networks," *Neural Computation*, vol. 5, no. 6, pp. 976–990, 1993.
- [11] M. W. Goudreau, C. L. Giles, S. T. Chakradhar, and D. Chen, "First-order versus second-order single-layer recurrent neural networks," *IEEE Transactions on Neural Networks*, vol. 5, no. 3, pp. 511–513, 1994.
- [12] R. Alquézar and A. Sanfeliu, "An algebraic framework to represent finite state machines in single-layer recurrent neural networks," *Neural Computation*, vol. 7, no. 5, pp. 931–949, 1995.
- [13] P. Frasconi, M. Gori, M. Maggini, and G. Soda, "Unified integration of explicit knowledge and learning by example in recurrent networks," *IEEE Trans. Knowl. Data Eng.*, vol. 7, no. 2, pp. 340–346, 1995.
- [14] S. C. Kremer, "On the computational power of elman-style recurrent networks," *Neural Networks, IEEE Transactions on*, vol. 6, no. 4, pp. 1000–1004, 1995.
- [15] P. Frasconi, M. Gori, M. Maggini, and G. Soda, "Representation of finite state automata in recurrent radial basis function networks," *Machine Learning*, vol. 23, no. 1, pp. 5–32, 1996.
- [16] B. G. Horne and D. R. Hush, "Bounds on the complexity of recurrent neural network implementations of finite state machines," *Neural Networks*, vol. 9, no. 2, pp. 243–252, 1996.
- [17] C. W. Omlin and C. L. Giles, "Stable encoding of large finite-state automata in recurrent neural networks with sigmoid discriminants," *Neural Computation*, vol. 8, no. 4, pp. 675–696, 1996.
- [18] —, "Constructing deterministic finite-state automata in recurrent neural networks," *J. ACM*, vol. 43, no. 6, pp. 937–972, 1996.
- [19] H. T. Siegelmann, "Recurrent neural networks and finite automata," *Computational Intelligence*, vol. 12, pp. 567–574, 1996.
- [20] A. M. Turing, "Intelligent machinery," National Physical Laboratory, Teddington, UK, Technical Report, 1948.
- [21] J. B. Pollack, "On connectionist models of natural language processing," Ph.D. dissertation, Computing Research Laboratory, New Mexico State University, Las Cruces, NM, 1987.
- [22] R. Hartley and H. Szu, "A comparison of the computational power of neural network models," in *Proceedings of the IEEE First International Conference on Neural Networks*, C. Butler, Ed. IEEE, 1987, pp. 17–22.
- [23] H. T. Siegelmann and E. D. Sontag, "On the computational power of neural nets," *J. Comput. Syst. Sci.*, vol. 50, no. 1, pp. 132–150, 1995.
- [24] J. Kilian and H. T. Siegelmann, "The dynamic universality of sigmoidal neural networks," *Inf. Comput.*, vol. 128, no. 1, pp. 48–56, 1996.
- [25] H. Hyötyniemi, "Turing machines are recurrent neural networks," in *sTeP '96 - Genes, Nets and Symbols; Finnish Artificial Intelligence Conference, Vaasa 20-23 Aug. 1996*, J. Alander, T. Honkela, and J. M., Eds. Vaasa, Finland: University of Vaasa, Finnish Artificial Intelligence Society (FAIS), 1996, pp. 13–24, *sTeP '96 - Genes, Nets and Symbols; Finnish Artificial Intelligence Conference, Vaasa, Finland, 20-23 August 1996*.
- [26] J. a. P. G. Neto, H. T. Siegelmann, J. F. Costa, and C. P. S. Araujo, "Turing universality of neural nets (revisited)," in *EUROCAST '97: Proceedings of the A Selection of Papers from the 6th International Workshop on Computer Aided Systems Theory*. London, UK: Springer-Verlag, 1997, pp. 361–366.
- [27] H. T. Siegelmann and E. D. Sontag, "Analog computation via neural networks," *Theor. Comput. Sci.*, vol. 131, no. 2, pp. 331–360, 1994.
- [28] J. L. Balcázar, R. Gavaldà, and H. T. Siegelmann, "Computational power of neural networks: a characterization in terms of kolmogorov complexity," *IEEE Transactions on Information Theory*, vol. 43, no. 4, pp. 1175–1183, 1997.
- [29] H. T. Siegelmann, "Neural and super-Turing computing," *Minds Mach.*, vol. 13, no. 1, pp. 103–114, 2003.
- [30] J. Cabessa and H. T. Siegelmann, "Evolving recurrent neural networks are super-Turing," in *Proceedings of IJCNN 2011*. IEEE, 2011, pp. 3200–3206.
- [31] —, "The super-Turing computational power of plastic recurrent neural networks," *Int. J. Neural Syst.*, vol. 24, no. 8, 2014.
- [32] —, "The computational power of interactive recurrent neural networks," *Neural Computation*, vol. 24, no. 4, pp. 996–1019, 2012.
- [33] J. Cabessa and A. E. P. Villa, "The expressive power of analog recurrent neural networks on infinite input streams," *Theor. Comput. Sci.*, vol. 436, pp. 23–34, 2012.
- [34] —, "An attractor-based complexity measurement for boolean recurrent neural networks," *PLoS ONE*, vol. 9, no. 4, pp. e94204+, 2014.
- [35] —, "Recurrent neural networks and super-Turing interactive computation," in *Artificial Neural Networks: Methods and Applications in Bio-/Neuroinformatics*, P. Koprivkova-Hristova, V. Mladenov, and K. N. Kasabov, Eds. Springer, 2015, pp. 1–29.
- [36] J. Cabessa and J. Duparc, "Expressive power of nondeterministic recurrent neural networks in terms of their attractor dynamics," *IJUC*, vol. 12, no. 1, pp. 25–50, 2016.
- [37] J. Cabessa and A. E. P. Villa, "Expressive power of first-order recurrent neural networks determined by their attractor dynamics," *Journal of Computer and System Sciences*, vol. 82, no. 8, pp. 1232–1250, 2016.
- [38] F. C. Hoppensteadt and E. M. Izhikevich, "Synchronization of laser oscillators, associative memory, and optical neurocomputing," *Phys. Rev. E*, vol. 62, pp. 4010–4013, Sep 2000.
- [39] D. Xu, J. C. Principe, and J. G. Harris, "Logic computation using coupled neural oscillators," in *Circuits and Systems, 2004. ISCAS '04. Proceedings of the 2004 International Symposium on*, vol. 5, May 2004, pp. V-788–V-791 Vol.5.
- [40] M. Zanin, F. Del Pozo, and S. Boccaletti, "Computation emerges from adaptive synchronization of networking neurons," *PLoS ONE*, vol. 6, no. 11, pp. 1–6, 11 2011.
- [41] D. Malagarriga, M. A. García-Vellisca, A. E. P. Villa, J. M. Buldú, J. García-Ojalvo, and A. J. P. Rivero, "Synchronization-based computation through networks of coupled oscillators," *Front. Comput. Neurosci.*, vol. 9, no. 97, 2015.
- [42] O. Feinerman, A. Rotem, and E. Moses, "Reliable neuronal logic devices from patterned hippocampal cultures," *Nature Physics*, vol. 4, pp. 967–973, 10 2008.
- [43] F. Wolf and T. Geisel, "Neurophysics: Logic gates come to life," *Nat. Phys.*, vol. 4, no. 12, pp. 905–906, 2008.
- [44] M. Abeles, *Local Cortical Circuits. An Electrophysiological Study*, ser. Studies of Brain Function. Berlin Heidelberg New York: Springer-Verlag, 1982, vol. 6.
- [45] —, *Corticonics: Neuronal Circuits of the Cerebral Cortex*, 1st ed. Cambridge University Press, 1991.
- [46] —, "Time is precious," *Science*, vol. 304, no. 5670, pp. 523–524, 2004.
- [47] Y. Ikegaya, G. Aaron, R. Cossart, D. Aronov, I. Lampl, D. Ferster, and R. Yuste, "Synfire chains and cortical songs: Temporal modules of cortical activity," *Science*, vol. 304, no. 5670, pp. 559–564, 2004.
- [48] Z. Mainen and T. Sejnowski, "Reliability of spike timing in neocortical neurons," *Science*, vol. 268, no. 5216, pp. 1503–1506, 1995.
- [49] P. Zheng and J. Triesch, "Robust development of synfire chains from multiple plasticity mechanisms," *Front. Comput. Neurosci.*, vol. 8, no. 66, 2014.
- [50] J. Cabessa and P. Masulli, "Emulation of finite state automata with networks of synfire rings," in *2016 International Joint Conference on Neural Networks, IJCNN 2016, Anchorage, AK, USA, May 14-19, 2017*. IEEE, 2017, pp. 4641–4648.
- [51] J. Cabessa, G. Horcholle-Bossavit, and B. Quenet, "Neural computation with spiking neural networks composed of synfire rings," in *Artificial Neural Networks and Machine Learning - ICANN 2017 - 26th International Conference on Artificial Neural Networks, Alghero, Sardinia, Italy, September 11-14, 2017, Proceedings*, ser. Lecture Notes in Computer Science, A. E. P. Villa, P. Masulli, and A. J. P. Rivero, Eds. Springer, 2017, pp. Accepted, to appear.
- [52] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *The Journal of Physiology*, vol. 117, no. 4, pp. 500–544, 1952.
- [53] A. Tchaptchet, S. Postnova, C. Finke, H. Schneider, M. T. Huber, and H. A. Braun, "Modeling neuronal activity in relation to experimental voltage-/patch-clamp recordings," *Brain Research*, vol. 1536, pp. 159–167, 2013, selected papers presented at the Tenth International Neural Coding Workshop, Prague, Czech Republic, 2012.