A Neural Nanonetwork Model Based on Cell Signaling Molecules

Áron Szabó*†, Gábor Vattay*§ and Dániel Kondor*§
*Department of Physics of Complex Systems
Eötvös University, H-1117 Budapest, Pázmány P. s. 1/A, Hungary
† Email: szaboahun@gmail.com ‡ Email: vattay@elte.hu § Email: kondor.dani@gmail.com

***Abstract*—All cells have to adapt to changing chemical environments. The signaling system reacts to external molecular 'inputs' arriving at the receptors by activating cellular responses via transcription factors generating proper proteins as 'outputs'. The signal transduction network connecting inputs and outputs acts as a molecular computer mimicking a neural network, a 'chemical brain' of the cell. The dynamics of concentrations of various signal proteins in the cell are described by continuous kinetic models proposed recently. In this paper we introduce a special neural network model based on the ordinary differential equations of the kinetic processes. We show that supervised learning can be implemented using the delta rule for updating the weights of the molecular neurons. We demonstrate the concept by realizing some of the basic logical gates in the model.**

I. Introduction

All living entities have to adapt to their environment. This is particularly true for unicellular organisms. In this case, environmental signals are quite simple, eg. temperature changes or changes in the chemical composition of the environment. Their so-called signaling network which consists of thousands of proteins reacts to such external stimuli. Its role is similar to the neural network of higher organisms. The high degree of interconnectedness and complexity of the protein network makes it very similar to a neural network. The nodes of this network are the proteins that travel in the cell's inner volume by diffusion.

Proteins in the signaling network can be regarded as information processing units: they produce molecules according to their molecular input. These proteins have two states: an activated and a deactivated. A protein in its activated state is also called phosphorylated. Hence this two-state system can be regarded as a binary bit based information storage system and passage of the activated state from protein to protein can be regarded as information propagation in the network. Such properties of the cellular signaling network enable us to use this system for computations and also as a nanoscale communication network.

Recently, it has been shown [1] that cell based molecular nano-communication networks can be modeled in terms of information theory. The receptor of the cell membrane (ligand) can be regarded as a sender and the cellular nucleus can be regarded as a receiver of the intra-cell communication. The response of the nucleus for the incoming cellular signals can be the initiation of the transcription of a specific gene. The transcription factors are the output nodes of the cell signaling network. Linear network coding [2] is one of the ways to formulate the signaling pathway network. The phosphorylation/dephosphorylation process can be represented by a network coding model [1]. Control of engineered molecular motors via signaling pathways [3], [4] is also possible [1] making possible to develop cellular systems which respond to external stimuli with various mechanical or electrical actions.

The realization of a deterministic process based programming of cellular communication systems requires further advances in molecular technology. Yet, logical gates operate in vivo cellular systems: the design logic of cannabionoid receptor signaling network has recently been uncovered [5]. In living cells the information flow is carried by a swarm of activated protein molecules and information passage between proteins is described by reaction-diffusion equations governing the concentrations of macroscopic number of molecules. Such computational systems – based on the concentrations of signaling protein molecules – has been envisioned by Bray [6] in 1995.

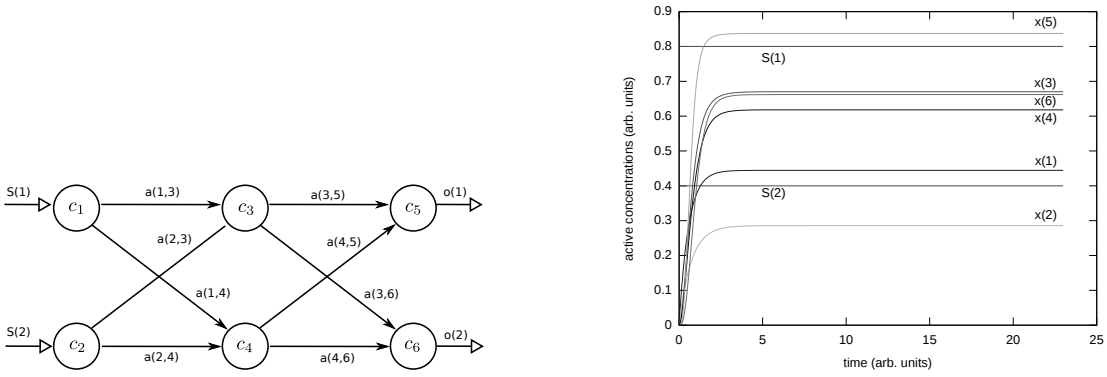
In this paper we show that the mass action kinetics of cell signaling networks can be designed and trained to carry out logical calculations. Elements of training algorithms developed in machine learning such as 'delta rule' and 'feed forward networks' can be realized.

The article is organized the following way. In section II we introduce a mass action kinetic model of the signaling network. In section III we deduce a learning algorithm for tuning the parameters of the network. Finally, in section IV we present some of the basic logic gates to illustrate the concept.

II. The model

Hereby we introduce a simple model of the network described in the Introduction. We assume that the distribution of protein concentrations in the system is spatially homogeneous and hence we can treat the problem with ordinary differential equations in the framework of mass action kinetics like in [7]. Let us assume the overall concentration of a protein including activated and deactivated parts is constant, c_i for the ith protein. The concentration of the ith protein's activated part is denoted by x_i, the deactivated part's concentration is $c_i - x_i$ accordingly. We assume, following Kartal and Ebenhöh [7], that the change of protein concentrations is described by the following equations:

978-1-4244-9920-5/11/$26.00 ©2011 IEEE



(a) (b)

Fig. 1. (a) A 2–2–2 network. In this case, $n = 6$ and $M = 2$. See section IV for explanation. (b) Relaxation of the 2–2–2 network. In the picture, $S_1 = 0.8$, $S_2 = 0.4$, and all proteins were totally inactive initially. We can observe that the network reaches equilibrium fast.

$$\frac{\mathrm{d}x_i}{\mathrm{d}t} = C_i S_i (c_i - x_i) + \sum_j x_j a_{ji} (c_i - x_i) - b_i x_i. \qquad (1)$$

The first term stands for the extracellular signal S_i activating the deactivated part of the ith protein with a coefficient of proportionality C_i. The second term describes the activation caused by the other proteins in the network. By matrix a_{ji}, we take into account that these processes have different probabilities. We can regard this network as a graph, where a_{ij} is the weighted adjacency matrix. The last term is due to the spontaneous deactivation of the protein with a characteristic time $1/b_i$.

Numerical simulations show (see Fig. 1b) that the system relaxes fast to an equilibrium point, which entitles us to study the equilibrium solution of the differential equation only. This means we have to solve the following algebraic equations:

$$0 = C_i S_i (c_i - x_i) + \sum_j x_j a_{ji} (c_i - x_i) - b_i x_i. \qquad (2)$$

Let us define the net input of the ith protein by $N_i := \sum_j x_j a_{ji} + C_i S_i$. It can be easily shown that (2) is equivalent to the implicit equation

$$x_i = c_i \frac{N_i}{N_i + b_i} = f_i(N_i), \qquad (3)$$

where $f_i(x) = c_i x/(x + b_i)$ is the so-called activation function for the ith protein. Observe that $f_i(x)$ is a monotonically increasing function, and it also has an upper bound c_i. In most artificial neural networks, the activation function is chosen to be a sigmoid function. Equation (3) can be solved numerically by iteration. Note that the right hand side of (3) is independent of x_i if we exclude self-interaction of proteins (i.e. $a_{ii} \equiv 0$).

III. The teaching algorithm

In this section we are going to examine the learning abilities of the previously introduced model. We are going to deal only with the so-called supervised learning. This means we want our network to give a desired response to a given input. In other words, we want our system to learn input–output patterns. In order to measure the quality of learning, we introduce the quadratic error function:

$$E := \sum_i (x_i - o_i)^2, \qquad (4)$$

where o_i is the desired output for the ith protein and the sum is executed only for the output proteins, where we require a given output o_i. The rest of the protein concentrations are not prescribed. Our goal is to minimize this error function by tuning the parameters of the system.

One way of minimizing this function is the so-called delta rule which is a steepest descent method. We introduce a small λ step parameter and in every teaching step we move along the gradient of the error function given by

$$p^{new} = p^{old} - \lambda \frac{\partial E}{\partial p}, \qquad (5)$$

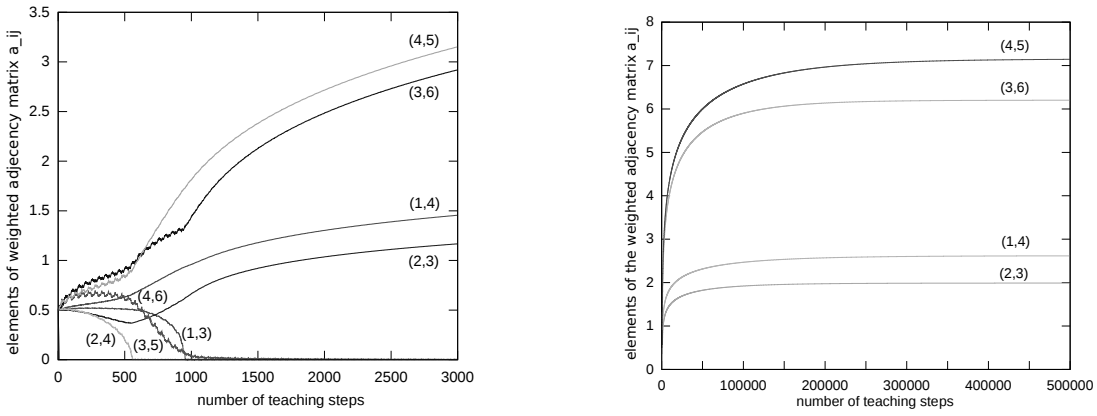
where p is an arbitrary parameter (i.e. overall concentration, time constant, or weighted adjacency matrix element). However, attention must be paid to the fact that only certain parameter values have a physical relevance (eg. negative concentrations are meaningless). We should check after every step whether the parameter values are valid.

Let there be n proteins in the network. Let there be M output proteins. Let us index the proteins so that the last M ones are the outputs. Using (3) and the chain rule recursively, it can be easily shown that we have explicit and easily computable formulae for the change of the parameters. For the change of the maximal concentrations, we have

$$c_m \mapsto c_m + \lambda_c \sum_{i=0}^{M-1} (o_{M-i} - x_{n-i})\, t_m \left[(Id - J)^{-1}\right]_{n-i,m}, \qquad (6)$$

where λ_c is the step parameter for the overall concentrations, Id is the $n \times n$ identity matrix, J is the Jacobian of (3):

$$J_{ij} = \frac{\partial f_i}{\partial x_j} = \frac{c_i b_i}{(N_i + b_i)^2} a_{ji}, \qquad (7)$$



(a) (b)

Fig. 2. Teaching of the identity function. In the figures, the matrix elements of the weighted adjacency matrix are plotted against the number of teaching steps. Pay attention to the different scaling of the axes. We denote the matrix elements by their indices in brackets. Fig. (a) shows the initial transient phase of learning, while Fig. (b) gives us a broader look over the learning process. The parameter values were: $c_i \equiv 1$, the b_i values were randomly selected from the $(0, 1)$ interval. Only the a_{ij} values were changing ($\lambda_a = 0.1$), the step parameters for the other parameters were zero. See section IV-A for a more detailed description.

and

$$t_i = 1 - \frac{b_i}{N_i + b_i} \tag{8}$$

is an auxiliary vector. A similar updating rule holds for the weighted adjacency matrix:

$$a_{lm} \mapsto a_{lm} + \lambda_a \sum_{i=0}^{M-1} (o_{M-i} - x_{n-i})\, x_l \left[(Id - J)^{-1} K\right]_{n-i,m}, \tag{9}$$

where λ_a is the step parameter for the adjacency matrix, and

$$K_{im} = \frac{c_i b_i}{(N_i + b_i)^2} \delta_{im} \tag{10}$$

is an auxiliary matrix. We can deduce another updating rule for the inverse characteristic times (b_i).

$$b_m \mapsto b_m + \lambda_b \sum_{i=0}^{M-1} (o_{M-i} - x_{n-i})\, q_m \left[(Id - J)^{-1}\right]_{n-i,m}, \tag{11}$$

where λ_b is the step parameter for the inverse characteristic times, and we introduced an auxilliary vector

$$q_i = -\frac{c_i N_i}{(N_i + b_i)^2}. \tag{12}$$

Observe that all variables in (6), (9), and (11) – excluding the step values λ_a, λ_b, and λ_c – have to be updated in every teaching step since they depend on the actual concentration values x_i.

This algorithm is highly sensitive to the step parameter λ. If we choose a too large step size we may never reach a minimum. On the other hand small step size implies slow convergence. Another problem is the fact that the Delta Rule can be stuck in local minima. Another interesting fact is that the parameter values depend not only on the given input–output patterns but on the *order* of the patterns. Nonetheless, this method works successfully in our numerical examples shown next.

A biophysical interpretation of the temporal evolution of the parameters during learning is also feasible. The overall concentracion changes due to the transcription factors (TFs) at the output of the network because the TFs make the nucleus produce the proteins that enter the network and hence the overall concentrations change. The change of the weighted adjacency matrix elements and that of the characteristic times can be due to conformal changes in the proteins or (in a rarer case) due to evolutionary mutations.

IV. Simulations Concerning Logic Gates

According to the model and the teaching algorithm described in the previous sections, numerical simulations were carried out. Our goal was to illustrate the abilities of this simple system by realizing some of the basic logic gates. We chose three logical fuctions for this purpose: identity, disjunction and exclusive disjunction (see Table I for the truth tables). We built a simple 'minimal' network for the simulations. It consists of layers with two proteins in each. Before training, all proteins interact with the proteins on the next layer. We are going to refer these networks by the number of proteins in the layers, eg. 2–2–2 refers to a network with three layers and there are two proteins in each layer (see Fig. 1a).

We used $o_i = 0$ and $o_i = 0.9c_i$ for representing the logical false and true values, respectively. Note that we cannot prescribe c_i for the truth value because $x_i < c_i$ according to (3). We built teaching patterns according to the truth table of the given logic functions. Since the delta rule is sensitive to the order of the patterns we used random input truth values (i.e. $S_i = 0, 1$).

A teaching step consists of two phases. First, the network's response to the actual pattern's external inputs (S_i) is computed by (3). Second, the parameters are updated according to (6), (9), and (11).

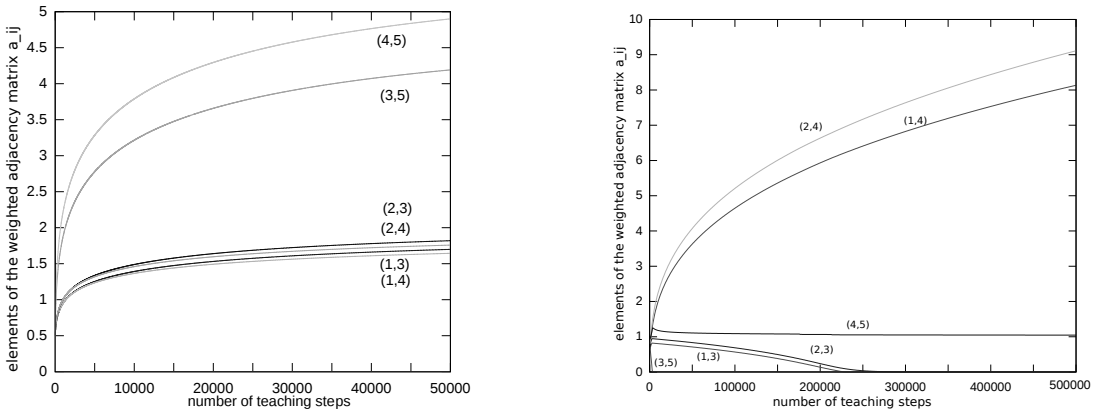


Fig. 3. Teaching of the disjunction and the exclusive disjunction. The a_{ij} values are plotted against the number of teaching steps. We denote the matrix elements by their indices in brackets. The parameter values were: $c_i \equiv 1$, the b_i values were randomly selected from the $(0, 1)$ interval. Only the a_{ij} values were changing ($\lambda_a = 0.1$), the step parameters for the other parameters were zero. Fig. (a) shows the disjunction function (see section. IV-B). Note the absence of the transient phase. Fig. (b) shows the exclusive disjunction function (see section. IV-C). The error is high in this case and doesn't decrease significantly during the training.

A. Identity

First we trained the network for the two-variable identity function. The simplest network for this function is a 2–2 network. The simulation showed that changing of the overall concentrations or the characteristic times is insufficient for a low error level. The network is trainable by modifying the weighted adjacency matrix a_{ij} (see Fig. 2a). We can see in the figure that the initial adjacency matrix is far from optimal but the training is efficient. The optimal choice for the network topology (and thus for the adjacency matrix) would be two separate pathways connecting the input to the corresponding output. This means that some of the a_{ij} elements have to be zero. We can see a transient phase in the figure, in which the the 'unwanted' connections disappear, and the error is big. The matrix elements cannot decrease under zero because this would have no physical meaning and we take this fact into account during the simulations. After the first transient phase, the fine-tuning of the non-zero matrix elements take place. The error decreases nearly exponentially in this phase. We can see in Fig. 2b that the matrix elements saturate after a sufficiently large amount of teaching steps. The final error after 500,000 steps is in the order of 10^{-8}.

The initial transient phase is longer in larger networks since the network has to 'select' a path configuration out of equally satisfactory alternatives. The behaviour is similar to the 2–2 case in the post transient phase.

B. Disjunction

The next simple logical function to present is the logical disjunction (OR). Since this is a single-valued function, we added one more layer to the 2–2 network to get a 2–2–1 network. According to the simulations, there is no transient phase like in the previous case, and the error is nearly exponentially decreasing during the whole teaching process. The lack of the transient phase may indicate that the network's topology is suitable for this pattern. Furthermore, in contrast to the identity pattern, learning by changing the overall concentrations is comparable effective to the one by changing the weighted adjacency matrix. We may say that the learning process via the overall concentrations is possible if the topology of the network, i.e. the weighted adjacency matrix, is suitable for the pattern. In other words, the overall concentrations can only fine-tune this system. The final error after 500,000 steps is in the order of 10^{-4} for the change of c_is and for the change of a_{ij}s.

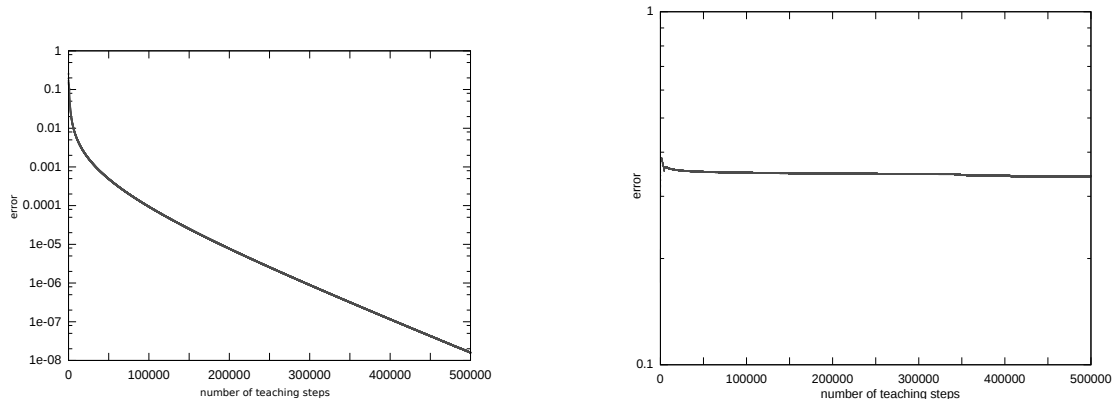
C. Exclusive disjunction

Despite the success in learning the identity and the disjunction, our simple network cannot learn every pattern. If we build a training pattern set form the exclusive disjunction (XOR), we find that the 2–2–1 network is incapable of realizing the (true,true) $\mapsto$ false assignment because the corresponding output value is larger than any of the other outputs. The error after 500,000 steps is in the order of 0.1, and doesn't change significantly during the training (see Fig. 4b).

A learnable pattern is monotonic in the following sense: 1. if the input is zero in all components, the output is zero as well. 2. If some component of the input increases, the output cannot decrease. This is a consequence of the monotonicity of the activation funcion f, and – more importantly – the non-negativity of the elements of the weighted adjacency matrix. Because of these two properties, an increase of an input causes increase in the activation of the neighbouring proteins, which causes increase in the activation of their neighbours, etc. So this increase 'propagates' through the network – including the output proteins as well. It can be easily shown that the identity and the conjunction are monotonic but exclusive disjunction is not.

V. Summary, Conclusions and Acknowledgements

We presented a quantitative model for the cell's signaling network. Considering the fast relaxation, we gave a method



(a) (b)

Fig. 4. Quadratic error during the training. In the figures, the quadratic error is plotted against the number of teaching steps. Pay attention to the different scaling of the axes. Fig. (a) shows the plot corresponding to the 2–2 identity network, while in Fig. (b) we can see the plot for the 2–2–1 XOR. It can be seen that the error exponentially decreases in the 'identity case'. On the other hand, teaching is insufficent in the 'XOR case'. See sections IV-A and IV-C for a more detailed description.

S_1	S_2	identity	OR	XOR
F	F	F,F	F	F
F	T	F,T	T	T
T	F	T,F	T	T
T	T	T,T	T	**F**

TABLE I
TRUTH TABLES FOR THE LOGIC GATES USED IN THE PAPER. 'F' DENOTES LOGICAL FALSE AND 'T' DENOTES LOGICAL TRUE. MONOTONICITY FAILS IN THE BOLD CASE.

for determining the equilibrious concentration values of the proteins. We presented a teaching method as well for the changes in the parameter values in the network according to the delta rule. We realized some basic logic functions on a simple network to show the properties of this model. We observed that the change of the interaction between the proteins is the main effect in the learning process of the network, the overall concentration of the proteins and the characteristic time for deactivation can only fine-tune the network. We gave a necessary condition for the learnable patterns. The cause of this condition seems to be that this model treats the activating protein–protein connections only (due to the non-negativity of the weighted adjacency matrix).

Our simple networks also had neither loops nor inhibitory protein–protein interactions. If we allow loops in the network, there will be nonzero solutions of (3). [7] This means that a loop can act like an inner source and hence the first condition of monotonicity can be relaxed. Inhibitory interactions can modify the monotonous activation functions and a great variety of functions including the sigmoid becomes possible. [8] Further investigations are required to clarify the exact role of the inhibitory effects in the trainability of the signaling network.

This paper is a part of a Project supported by the European Union and co-financed by the European Social Fund (grant agreement no. TÁMOP 4.2.1./B-09/1/KMR-2010-0003).

REFERENCES

[1] J.-Q. Liu, "Molecular informatics of nano-communication based on cells: A brief survey," *Nano Communication Networks*, vol. 1, pp. 118–125, June 2010.

[2] S. Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Transactions on Information Theory*, vol. 49, no. 2, pp. 371–381, February 2003.

[3] K. J. Verhey and T. A. Rapoport, "Kinesin carries the signal," *Trends Biochem Sci*, vol. 26, no. 9, pp. 545–550, September 2001.

[4] D. Horiuchi, C. A. Collins, P. Bhat, R. V. Barkus, A. DiAntonio, and W. M. Saxton, "Control of a kinesin-cargo linkage mechanism by JNK pathway kinases," *Current Biology*, vol. 17, no. 15, pp. 1313–1317, 2007.

[5] K. D. Bromberg, A. Ma'ayan, S. R. Neves, and R. Iyengar, "Design logic of a cannabinoid receptor signaling network that triggers neurite outgrowth." *Science (New York, N.Y.)*, vol. 320, no. 5878, pp. 903–909, May 2008.

[6] D. Bray, "Protein molecules as computational elements in living cells," *Nature*, vol. 376, no. 6538, pp. 307–312, July 1995.

[7] O. Kartal and O. Ebenhöh, "Ground state robustness as an evolutionary design principle in signaling networks," *PLoS ONE*, vol. 4, no. 12, December 2009.

[8] J. J. Tyson, K. C. Chen, and B. Novák, "Sniffers, buzzers, toggles and blinkers: dynamics of regulatory and signaling pathways in the cell," *Current Opinion in Cell Biology*, vol. 15, no. 2, pp. 221–231, 2003.