NeuroSim: Neuronal Simulation System of biological neural networks

Olga Kroupina

October 9, 2003

1 Introduction

Neuroscientists use computer simulations of neural systems in their efforts to understand processes that underlie neural function. As experimental data increase, it becomes clear that detailed physiological data alone are not enough to infer how neural circuits work. Experimentalists appear to be recognizing the need for a quantitative approach to the exploration of the functional consequences of particular neural features, which is provided by modelling. The number of computer simulation programs is designed as a tool for development and simulation of realistic models of single neurons and neural networks.

The present available packages for modelling of biological neural networks are often dedicated Unix-based simulation packages, which require rather large computational power from workstations, typically Unix systems. The widely distributed packages, as Genesis [8] and Neuron [4], have their own interpreted scripting language, in which users define components and running parameters for their simulations. In the hands of experienced users with access to a compatible computer system, these modelling packages are powerful research tools. However, they do suffer several drawbacks for non-expert users: they don't provide a Graphical User Interface (GUI) or have a very simple one, and as a result of it they can't visually represent the simulation process. Also, the formal structure of the language is difficult and time consuming to learn; at least initial knowledge and skills about Unix system are necessary for users.

2 The problem and its solution

Large computational power is necessary for solving the system of the differential equations which describe the spread and interaction of electrical and chemical signals in neuronal networks. The Client-Server architecture of the application software is an alternative for performing intensive calculation on the high-power computers (workstations), while control and visual results presentation are left to a personal computer.

All of it stimulated the development of the software, with following distinctive features:

- platform independence;
- user friendly and intuitive GUI;
- a comprehensive library of standard models, on basics of that definition of the complicated neuron systems and its modification without any programming skills is possible and not time consuming;
- reserve the possibility for extension of this library for users; standard and extensible format of the model description, which ensure compatibility with different software packages;
- active utilization of visual teaching methods for presentation neuronal networks and excitation and inhibition processes in it on the basics of the simulations results.

Realization of these ideas to a great extent gives a stimulus to developing a simulation system, "NeuroSim". It was realized in Java, a web-enabled cross-platform language. All aspects of developing and running simulations are mediated through a user-friendly GUI and no programming skills are necessary. The model description and resulting data are specified in the standard XML model description language.

NeuroSim was designed with a Client-Server structure, which is effective for numerical integration. The remote computer is used for extensive calculations and the personal computers for models and results presentation. At the first stage of the project the Genesis-Simulator [1] was used as a server on the powerful computer under Unix system on it.

The Java-Client is used for the models definition, transferring data to server and starting calculations on it. After performing the simulations,

data are received by the Java-Client, which performs the presentation of simulation results. The resulting data, received from the Server, can be played back. This gives the possibility to visually present the behavior of the different parameters, describing a model dynamic. The action potential evolution and the spread of "spike" events, channel conductance and voltage curves help to understand the detailed physiology and structure of neural networks.

The neurons can be modelled with Hodgkin-Huxley-type conductance channels [5]. The neurons can be coupled with synaptically activated channels. For more detailed exploring the models with compartment structure can be simulated. The program allows choosing between several integration methods, from the crude explicit forward Euler method through highly stable implicit methods [6]. One can define the value and integration step size influencing accuracy and speed of integration.

3 The Client-Server connection

Once a Java program has been started and the experimental model has been defined, the Client-Server connection can be established. The connection mechanism was made with the Java software tool - Remote Method Invocation (RMI). The idea is that the whole distributing objects can communicate among Java application on multiple machines. The values are sent back and forth, so that neither the client nor server has to do anything explicit with input and output streams, no parsing is required. The conversion of model description from Java client to server and results of integration back was realized by Java serialization facility.

The neurons, their connections and inserting mechanisms in the model implements as a serializable objects, the necessary for integration properties will be sent in the XML format. The running server will parse this information to the Genesis code and then integration will be started.

The transformation of data in the XML format into Genesis code was made with the XSLT (XML style sheet), the standard part of Java 2 standard Edition. The benefit of this method is that one can define formatting rules for transformation a XML document. The transformer was generated to apply it to documents of model properties.

Genesis gives the results in the standard output stream. The data contain in every time step the time and the value received after integration. It will

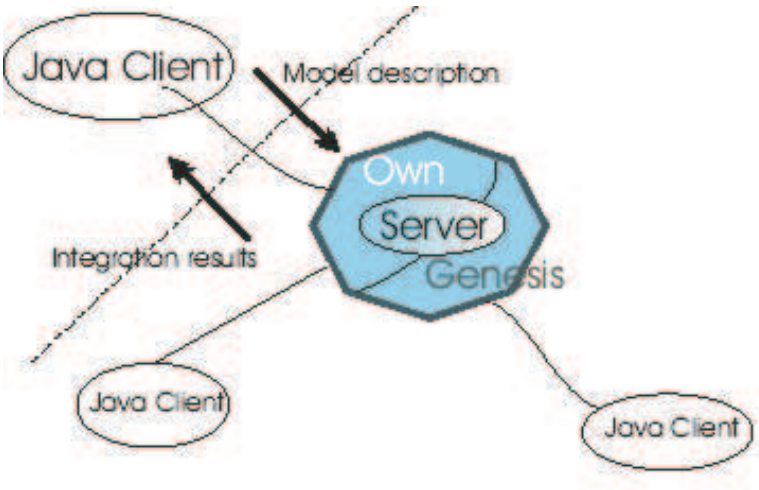

Figure 1: The Client-Server architecture.

be sent for the client and used for the simulation.

4 The Simulation

After the neurons were added and the properties of them were assigned the simulation can be started. For the real networks the data received from the remote server will be played back.

Java Client can represent the process of simulation, with the possibility to visually estimate the modelling process. The main panel with the neuron structures represents the spike events and the spread of the pulses between neurons. At the same time the graphs of the simulated parameters will be shown as well as their change in the time (See Fig. 2).

The windows "Waves", "Potentials" and "Conductances", which is available from menu "Window", are visually presented the process of simulation (See Fig. 3) . These windows contain two scrolling panels, the list of all elements with possibility to choose the observed graphs, and the resulting curves. The left scrolling panel has a pointer, which represent a time of simulation course. One can move it back and forth, so that the simulation will be started at the time, where the pointer is located. The time course of simulation the value is showed in the infoline. The detailed values of simu-

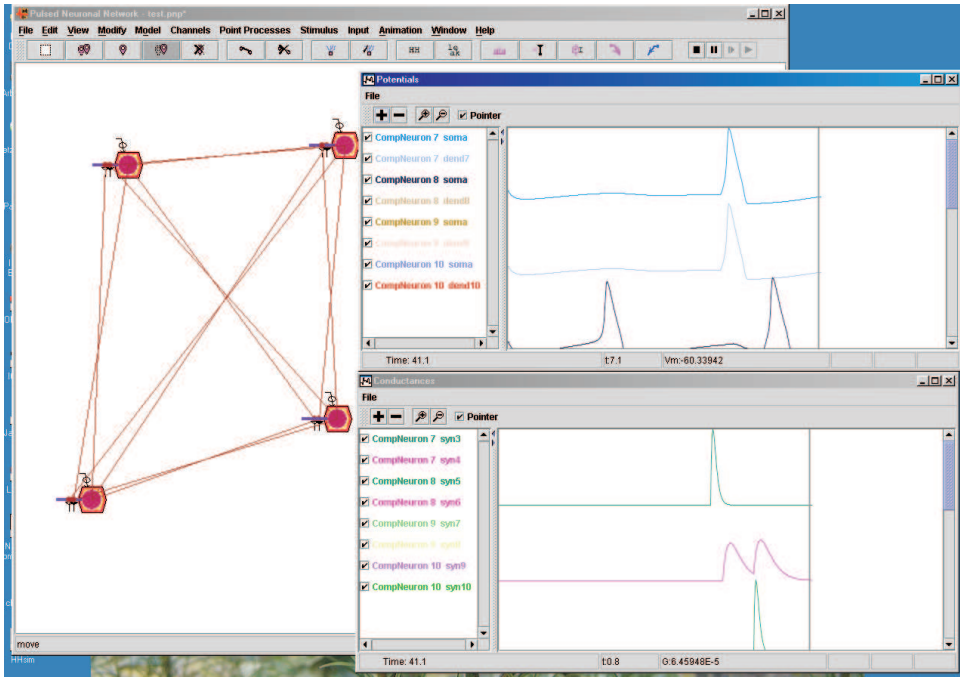

Figure 2: The resulting data.

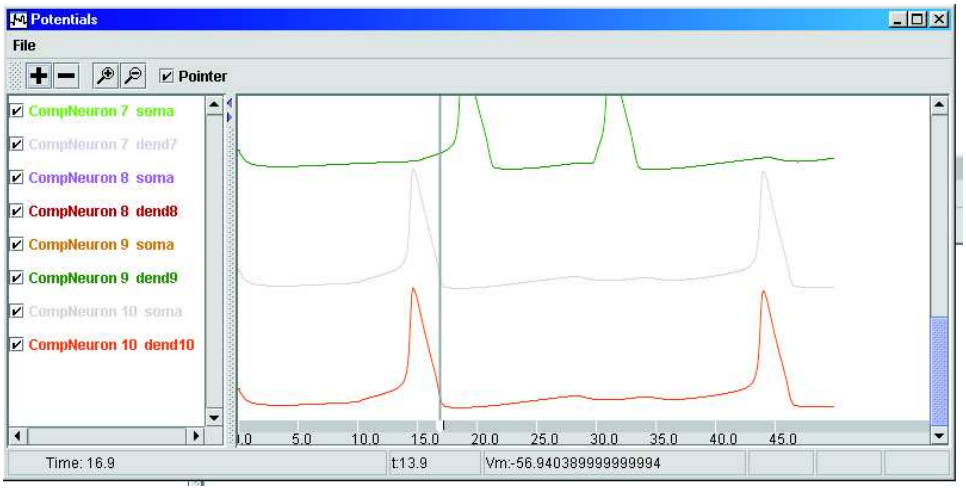

Figure 3: The window of potentials.

lated parameters are represented with the mouse movement on the graphical panels.

The process of simulation was implemented as a thread with two utility classes - SwingWorker(not swing class) and the Timer (javax.swing.Timer) classes. A SwingWorker object creates a thread to execute a time-consuming operation. Timers are used for performing a task either repeatedly or after a specified delay. After the operation is finished, SwingWorker gives you the option of executing some additional code in the event-dispatching thread. A SwingWorker moves a time-consuming task from an action event handler into a background thread, so that the GUI remains responsive. The Timer class allows you to schedule an arbitrary number of periodic or delayed actions with just one thread. Accordingly one can name these two modes as an asynchronous and synchronous.

Once the simulation has been set up, a main loop is begun. This loop makes one call to the library to simultaneously update all the simulation components, generates the output of interest - spike events and the movement of parameters curves, and repeats until done. The events are saved in the vector (OrderedDoubleVector) - priority queue, the data structure that stores "prioritized elements". The priority queue stores elements according to their priorities ("key" value), that is in our case a time of events arise, and supports removal and getting of elements only in order of priority. So,

the value of studied parameter together with the time of arising are saved in the OrderedDoubleVector.

5 Further work

The developed system for simulation of real neural networks has effective Client-Server structure. The Genesis simulator is used as a Server in the remote computer for numerical integration. In the second phase of the project the Genesis will be replaced with own Server for solving the systems of the differential equations describing the models.

At first, the realization of simulations based on compartment modelling is planed. It is explained by increasing interest in detailed models dependent on the anatomy and physiology of the neurons. The structures of cells divided into many isopotential compartments, joined by conductances, and activated with simulated ion channels and current injectors will be implemented. The ionic channels with Hodgkin-Huxley dynamic as often presented in neural models have to be simulated. On the basic of it the general voltage gated ionic channels will be derived [2]. The ion concentrations and its dynamic have to be taken into account. The connection of neurons will be made with chemical synapses.

The choice of numerical integration technique is a critical part of the simulation process. The most effective method for the general form of the differential equations arising in neural modelling is an exponential Euler method. This method together with another explicit methods gives accurate results for integration of simple cell models with few compartments. The implicit methods as a background Euler or implemented by Hines Crank-Nicholson will require a much smaller time step for simulation of multicompartmental models, because equation to be solved is stiff [3]. All groups of these methods have to be implemented.

Another direction of the project development is the integration NeuroSim with the electronic chalkboard, E-Chalk, a multimedia system for distance teaching [7]. NeuroSim is developed as a suitable system both for research and educational purposes. Together with the E-Chalk it can be used both as powerful education software and as the tools for research results presentation.

References

[1] J Bower and D. Beeman. *The Book of Genesis: Exploring Realistic Neural Models with the GEneral NEural SImulation System.* Springer TELOS, 1998.

[2] G.B. Ermentrout. Channeling with Bard. *Comp.Neuroscience*, Januar 1998.

[3] M. Hines. Effecient computation of branched nerve equations. *J.Biomed.Comp.*, 15:69–76, 1984.

[4] M. Hines. A program for simulation of nerve equations with branching geometries. *International Journal of Biomedical Computing*, 24:33–68, 1989.

[5] A. Hodgkin and A. Huxley. The components of membrane conductance in the giant axon of loligo. *J. Phyiol.*, 116:473–496, 1952.

[6] M. Mascagni. *Numerical methods for neuronal modelling.* MIT Press, Cambridge, Mass, 1989.

[7] R. Rojas, G. Friedland, L. Knipping, and W. L. Raffel. Elektronische Kreide: Eine Java-Multimedia-Tafel für den Präsenz- und Fernunterricht. Technical Report B-00-17, FU Berlin, Institut für Informatik, October 2000.

[8] Bhalla U.S. Uhley J.D. Wilson, M.A. and J.M. Bower. Genesis: a system for simulating neural networks. In *Advances in Neural Information Processing Systems.*, pages 348–353. Morgan Kaufman, San Mateo, 1989.