Proceedings of the 26th Annual International Conference of the IEEE EMBS
San Francisco, CA, USA • September 1-5, 2004

Hybrid neural networks – combining abstract and realistic neural units

William W. Lytton[1] & Michael Hines[2]

[1] SUNY Downstate, Brooklyn, NY 11203, Email: billl@neurosim.downstate.edu, Telephone:

(718) 270–1163 [2]Yale University, New Haven, CT, Email: michael.hines@yale.edu

Abstract— **There is a trade-off in neural network simulation between simulations that embody the details of neuronal biology and those that omit these details in favor of abstractions. The former approach appeals to physiologists and pharmacologists who can directly relate their experimental manipulations to parameter changes in the model. The latter approach appeals to physicists and mathematicians who seek analytic understanding of the behavior of large numbers of coupled simple units. This simplified approach is also valuable for practical reasons – a highly simplified unit will run several orders of magnitude faster than a complex, biologically realistic unit. In order to have our cake and eat it, we have developed hybrid networks in the Neuron simulator package. These make use of Neuron's local variable timestep method to permit simplified integrate-and-fire units to move ahead quickly while realistic neurons in the same network are integrated slowly.**

I. Introduction

The simulation load of a single multi-compartment neuron with realistic biophysical mechanisms can easily exceed the load associated with simulation of a several-thousand unit network of highly simplified abstract units. The multi-compartment neuron may have hundreds of compartments, each of which has several voltage-sensitive channels, each having their own intrinsic dynamics requiring ordinary-differential equation (ODE) solution. Such a simulation may have thousands of linked ODEs. Despite this computational difficulty, biophysically realistic simulation is of great value for experimentalists who can directly relate electrophysiological recordings and pharmacological manipulation to results obtained by simulation.

Because of these limitations, several simplifying approaches are commonly used to produce and simulate large neural networks. Commonly a large multi-compartment model will be reduced to a model with far fewer compartments for speed of simulation. A more aggressive approach is to maximally simplify the individual neuron, making it a single-state unit whose analogue state represents firing rate (the artificial neural-network approach). Alternately, the neuron may be represented as an integrate-and-fire (I&F) unit with a single state representing voltage and a fixed threshold which, when reached, enables spike production and synaptic output. Networks of these simplified units can be scaled to very large size, revealing emergent properties inherent in the extremely large networks of the brain. However, it is difficult to relate these simplified networks to the biophysical details measured experimentally.

II. Results

We are developing hybrid neuronal networks in order to combine some of the efficiencies associated with the simplifying approach with the experimental detail available in the biophysically-detailed approach. Development of these hybrid networks has required development of management and integration tools which have been instantiated in the Neuron simulator. The local variable time step method allows simplified I&F neurons to be integrated rapidly without having to wait for the slowly-integrating realistic neurons. We have also developed a relational database to organize wiring and identities of cells of several different classes and types.

A. Local variable time step method (lvardt)

Neuronal network simulation is intrinsically a hybrid task. On the one hand, biophysical mechanisms are dynamical systems requiring integration of ODEs continuously in time. On the other hand, all communications between the elements of the network are spike events, suggesting the use of an event driven simulation system. We have designed *lvardt* in order to allow the time-consuming ODE integration to be embedded within an

0-7803-8439-3/04/$20.00©2004 IEEE

event-driven framework. A side-effect of this hybrid design is the ability to efficiently simulate hybrid networks, with one or more biophysically-detailed neurons embedded in a network of highly simplified, event-driven units.

We have previously demonstrated some advantages of the global variable time step integrators CVODE and CVODES (Cohen 1994) over traditional fixed step methods such as Euler, Runge-Kutta or Crank-Nicholson for simulating single cells.(Hines & Carnevale 2001) The major advantage was due to the fact that neuron activity features spikes, requiring short times steps, followed by interspike intervals, which allow long time steps. The associated speed-up in the single-cell integration realm do not extend to simulation of networks however. A major reason is that the global timestep is governed by the fastest changing state-variable. In an active network, some cell is usually firing, requiring a small timestep for the entire network. Another, related reason, is that synaptic events generally cause a discontinuity in a parameter or state-variable. This requires a re-initialization as the integrator must start again with a new initial-condition problem. In a network simulation, this means re-initialization of the entire network due to a single state variable change in one cell. With re-initialization, the integrator is working without any past history. Hence the first step can only be first-order accurate and must be very short.

The critical problem in the implementation of the local time-step method (*lvardt*) is to ensure that when an event arrives at a cell at time t_e that all the state-variables for the receiving cell are also at time t_e. This requires coordinating individual integrators so that one cell does not get so far ahead that it cannot receive a synaptic signal from another cell.

lvardt integrates the network piece-meal, providing short time-step integration for active neurons and long time-step integration for inactive neurons, while maintaining consistent integration accuracy throughout. Neurons that fire at different times get their state-variables calculated at different times and, more importantly, different intervals. State-variables change quickly near threshold and at the peak of the action potential. At these times, the integrators use short timesteps to accurately follow the trajectory through state space. At other times, much larger dt's can be used to achieve the same accuracy for the more slowly varying state-variables. Using *lvardt*, a neuron that is inactive does not waste CPU time. Performance evaluation for a simple simulation of two single-compartment cells demonstrates that the global method integrates its 8 state-variables (the 4 Hodgkin-Huxley variables m, h, n, V for each cell) 177 times, while the integrators in the *lvardt* example integrate 4 state-variables 138 times (first-spike cell) and 115 times (second-spike cell) for a total of $4 \cdot (138 + 115) = 1012$ state-variable integrations. Comparison with the total integrations for the global method suggested a 40% speed-up which was confirmed by simulation.

The *lvardt* method creates a separate CVODES integrator for each cell in the network. Although there are many more integrators (n for n cells instead of 1), each integrator is more compact since it only has to handle the state-variables belonging to its particular neuron. Whether using one or n integrators, the total number of state-variables remains the same.

Because the system now is being calculated forward in time by multiple, independent integrators, an integration-coordinator is used to maintain the overall coherence of the integration. If the various neurons in the network are not connected, as in the case of testing parameter variation over a set of neurons, such coordination is not needed. However, in a network, the integration-coordinator is vital to permit synaptic signals to be communicated at appropriate times.

B. Organizing network wiring in a relational database

Real brain networks feature a bewildering array of neuron types, differing in dendritic shape and temporal response properties. The densities of each constituent neuron type is variable and the wiring between populations differs. Not only are there different neuron types, there are also different connection types with different signs (excitatory and inhibitory synapses) and different time courses (*e.g.,* $\mathrm{GABA_A}$ vs. $\mathrm{GABA_B}$ synapses). Such complexities make it difficult to design, implement and confirm the wiring diagram for a particular network. In designing a hybrid network, we add further complexity by representing some of these neurons with detailed multi-compartment models and the rest as simplified integrate-and-fire neurons.

In order to manage this complexity we have implemented relational databases in Neuron and developed a simplified query language in order to readily identify produce and identify connectivity. Each row of the database represents a connection between one presynaptic neuron and one postsynaptic neuron. We currently define a single connection with 14 columns. The presynaptic cell and postsynaptic cell are defined using 3 identifiers for each: biological cell type, cell ascension number, simulation type (integrate-and-fire or multi-compartment). 2 columns are used to define the relation of these cells to one another: the distance be-

tween the cells and the signal propagation delay between them (calculated using distance and axonal conduction properties determined by presynaptic cell type). At least 3 columns are needed to identify the synapse itself: connection weight (synaptic conductance strength), synaptic type (*e.g.,* AMPA, NMDA, $GABA_A$, $GABA_B$) and an identifier to directly access Neuron's internal synaptic representation. Optionally, another 3 columns are used to identify a secondary synaptic connection between the same two units (*e.g.,* colocalized $GABA_A$ and $GABA_B$ inhibitory connections).

Note that this particular database organization represents a set of compromise choices for current purposes. The database is not in normal form. In order to simplify it to third normal form, we would want to put all of the cell information into a separate table and omit calculated values such as the value for axonal delay. We have not optimized the database in this way so as to obtain optimal speed for definition and read-out of the network structure.

The use of this structure makes it easy to set parameters for the network based on chosen cellular relations. For example, the following query will pull out connections between thalamocortical (TC) and superficial cortical pyramidal (SU) neurons that are between 200 and 500 microns in distance:

```
sp.select("PRE-ID",TC,"POST-ID",
SU,"DISTANCE","[]",200,500)
```

Then these connections can be set to particular synaptic weights:

```
sp.fill("WT0",0.25,"WT1",0.1)
```

Here, WT0 represents the AMPA weight and WT1 represents the NMDA weight. Alternatively, a weight can be calculated based on distance, *e.g.,*:

```
sp.spr("WT0","<DIST>.c.apply
('falloff').mul(maxwt)")
```

Here, weight is based on distance after calculation using a fall-off function and multiplication by a maximum weight parameter.

In addition to network definition and weight setting, the database structure makes it easy to demonstrate connectivity graphically.

Hybrid networks

We are utilizing hybrid networks to explore two types of brain structures: a thalamocortical network with 4 cell types in a one-dimensional linear organization and a synfire chain network featuring sequential activation of groups of loosely connected excitatory and inhibitory cells.

Integrate-and-fire cells can be integrated quickly because the single state-variable describes a trajectory that can be calculated analytically. In the context of the Neuron event processor, this means that the state-variable can be updated based on its value at the time of the last event and the elapsed time since that event. After performing this calculation, the weight of an incoming synaptic event can be added to the state-variable to determine whether threshold has been reached and the cell should file. Similarly, in the case of an intrinsically active unit, an internal event time is generated following a spike, to signal the unit that it is time to spike again. Intervening synaptic arrivals will alter this intrinsic spike time, moving it forward in the case of excitation or backward in the case of inhibition.

In addition to these simple integrate-and-fire units, we are also developing units with more complex firing patterns. For example, an integrate-and-burst unit includes several state variables which determine inter- and intraburst adaptation as well as overall excitation. All of these state variables are independent and can be calculated analytically, allowing the same rapid, even-based simulation method.

III. Conclusion

The use of local integrators for individual neural network components allows for substantial efficiencies in handling both the ODES of realistic neuron simulation and the events produced by simplified artificial neurons. The ease of combination in turn suggests the development of hybrid networks to combine the accuracy of realistic simulation with the efficiency and scaling of artificial networks. Management of large networks of disparate elements – different types of neurons as well as different neuron implementations – suggest the use of databasing for network organization.

References

[1] S. Cohen and A. Hindmarsh, "Cvode user guide," Lawrence Livermore National Laboratory, Livermore, CA, Tech. Rep., 1994.

[2] M. Hines and N. Carnevale, "Neuron: a tool for neuroscientists," *The Neuroscientist*, vol. 7, pp. 123–135, 2001.