Why another simulator?

- Larger network models demand multicore computational tools. However, densely integrated network models, in which many nodes must update one another at every timestep (such as many biophysical models), are ill-suited for execution on computational clusters due to slow inter-node communication.
- Current general-purpose implementations are difficult to parallelize and require special coding to achieve limited multiprocessing capabilities. An ideal solution would separate the biological problems from the optimization problems.
- GPU hardware is a promising tool for portable computing due to its ability to hide inter-node communication delays.

Portable and Low Memory Python 3.x Metaclasses

Extend Myriad to a nonuniform memory access architecture to zero copy from simulation.

Type annotation support

Uses no memory leaks

Maximizes stack usage

Zero namespace collisions

Use “fat binary” option to simulator

Because of (myelinated sections, synaptic cleft explicitly as a separate compartment.

To adhere to these best practices, Myriad incorporates a separate “fat binary” option to increase compilation time but with lower binary overhead costs.

Granularity Enables Highly Flexible Model Design

(A) Neuron receiving inputs from two presynaptic neurons: one normal synapse from a myelinated axon and one specialized synapse (from an unmyelinated axon) in which the user wishes to model the synaptic cleft explicitly as a separate compartment.

(B) Presynaptic boutons Syaptic cleft

(B) Schematic example of Myriad compartments and mechanisms. Compartment and mechanism objects incorporate user-defined equations. A single extracellular space compartment here connects to all compartments except the synaptic cleft (multiple extracellular spaces also are supported). Adjacency is implemented as paired reciprocal mechanisms.

Implementation Details

- C99 standard-compliant
  - One-time portable kernels for x86_64
  - Upgrade path to C11
  - Supports clang GCC, ICC, possibly MCC

- <pthread.h> on CPU
  - Portable to low overhead costs
  - Upgrade path to C11

- <pthread.h> on GPU
  - Portable to low overhead costs
  - Upgrade path to C11

- Zero-copy from simulation

- Type annotation support

- Inheritance

- No namespace collisions

- Automatic type coercion

- No memory leaks

- Source in C, Python, in which users construct models using tools from

- Import support currently being investigated.

- Data export via Numpy

- Silver fast binary option to increase compilation time but with lower binary overhead costs.

- JIT Support for CUDA

- Use “fat binary” option to increase compilation time but with lower binary overhead costs.

- Fully-configurable compile-time options

Simulations Examples

(A) 100 Hodgkin-Huxley neurons coupled with inhibitory synapses to form an interneuron network game (ING) oscillatory network.

(B) Two Hodgkin-Huxley neurons, separated or connected by a gap junction

Example simulations coded at the implementation level in C

Planned Extensions

- Extend Myriad to a nonuniform memory access architecture to support multiple CUDA cards on a single high-speed bus.
- Implement simulator governor to run multiple instances in series or in parallel (e.g., on distributed architecture GPU clusters), to support parameter exploration and algorithmic optimization.
- Provide advanced users access to Myriad’s code generation API.
- Myriad is an arbitrarily programmable GPU-enabled computational framework that is in principle as appropriate for (e.g.) 3-D spatial diffusion models as for neuronal modeling. Assess Myriad’s utility for these different applications, and their synthesis.

Interested?

- Myriad is an open-source project that soon will be open for community participation.
- If you are interested in early-stage access as a contributor, please send a detailed email to both authors describing the reasons for your interest and your relevant skills in Python, C, and GPU coding as well as in neuroscience and related fields.
- If you are interested in beta testing as an end user, please send an email to both authors and/or sign up on the provided list.

References & Acknowledgments


http://www.gputechconf.com


This research was supported by an equipment grant from the NVIDIA Corporation.