Automated code generation of advanced plasticity rules for the SpiNNaker neuromorphic platform using NESTML

Charl Linssen(1,3), Pooja Babu(1,3), Andreas Rogalski(3,4), Bernhard Rumpe(3) and Abigail Morrison(2,3)

1. Simulation and Data Laboratory Neuroscience, Jülich Supercomputer Centre, Institute for Advanced Simulation, Jülich-Aachen Research Alliance, Forschungszentrum Jülich GmbH
2. Institute for Advanced Simulation IAS-6, Computational and Systems Neuroscience, Forschungszentrum Jülich GmbH
3. Software Engineering, Department of Computer Science 3, RWTH Aachen University, Germany
4. SpiNNcloud Systems GmbH, Dresden

Current neuromorphic and HPC workflows suffer from a usability gap, where mapping complex biological plasticity models to specialized hardware requires manual, error-prone, low-level coding. NESTML is a domain-specific modeling language that allows researchers in computational neuroscience to specify models of neurons and synapses in a precise and accessible way. These models can subsequently be used in dynamical simulations of spiking neural networks on various simulation platforms, such as NEST Simulator [2] or SpiNNaker [3]. This is achieved by means of platform-specific code generated by the NESTML toolchain. The generated code extends a simulation platform with new neuron models and synaptic plasticity rules, that can then be instantiated in a network of any size. Combining a user-friendly modeling language with automated code generation makes large-scale neural network simulation accessible to neuroscience researchers without requiring any prior training in computer science.

In this work, we establish an extension of the NESTML code generation toolchain that adds support for simulation of advanced synaptic plasticity rules on the SpiNNaker neuromorphic hardware platform [3]. We demonstrate our approach with code generation for a spike-timing dependent plasticity (STDP) synapse model in a simple network. The dynamics of the network is solved using exact integration [4]. The necessary numerical integration routines are automatically generated by the toolchain. We validate the simulation results by means of comparison of numerical outcomes of the simulation to those obtained from NEST Simulator running on a standard CPU as a reference. The proposed solution is generic, so that for instance the triplet STDP rule [5] can be expressed directly in the DSL, without requiring any further changes to the toolchain.

NESTML's features make models easier to write and maintain, as well as to share and reuse. It bolsters reproducibility in neuroscience by allowing results to be compared between platforms, while accelerating research and development for applications including edge computing and neuromorphic HPC. This project establishes the foundation for precise benchmarking of performance, memory efficiency, and power consumption. These will further benefit from the upcoming SpiNNaker-2 system installation at the Jülich Supercomputing Centre, making NESTML a key enabler in the transition towards modular supercomputing.

[1] https://github.com/nest/nestml/

[2] Gewaltig & Diesmann, Scholarpedia 2(4), 2007

[3] Furber et al., Proceedings of the IEEE 102(5), 2014

[4] Rotter and Diesmann, Biological Cybernetics 81, 1999

[5] Gjorgjieva, Clopath, Audet, & Pfister. Proc. Natl. Acad. Sci. U.S.A. 108 (48) 19383-19388,(2011).