Member of the Helmholtz Association

JÜLICH
FORSCHUNGSZENTRUM

The NEST neuronal network simulator:

Performance optimization techniques for high performance computing platforms

Alexander Peyser[†] and Wolfram Schenck[†‡]

[†]Simulation Lab Neuroscience – Bernstein Facility Simulation and Database Technology, Institute for Advanced Simulation, Jülich Aachen Research Alliance, Forschungszentrum Jülich, 52425 Jülich | Germany
a.peyser@fz-juelich.de

[‡]Department of Engineering Sciences and Mathematics
University of Applied Sciences Bielefeld
Bielefeld, Germany
wolfram.schenck@fh-bielefeld.de

Abstract

NEST (`http://www.nest-initiative.org`) is a spiking neural network simulator used in computational neuroscience to simulate interaction dynamics between neurons. It runs small networks on local machines and large brain-scale networks on the world's leading supercomputers. To reach both of these scales, NEST is hybrid-parallel, using OpenMP for shared memory parallelism and MPI to handle distributed memory parallelism. To extend simulations from short runs of 10^9 neurons toward long runs of 10^{11} neurons, increased performance is essential. That performance goal can only be achieved through a feedback loop between modeling of the software, profiling to identify bottlenecks, and improvement to the code-base.

HPCToolkit and SCORE-P toolkit were used to profile performance for a standard benchmark, the balanced Brunel network. We have additionally developed a performance model of the simulation stage of neural dynamics after network initialization and proxy code used to reduce the resources required to model production runs. We have pursued a semi-empirical approach by specifying a theoretical model with free parameters specified by fitting the model to empirical data. Thus we can extrapolate the scaling efficiency of NEST and by comparing components, identify algorithmic bottlenecks and performance issues which only show up at large simulation sizes.

Performance issues identified include: 1) buffering of random number generation lead to extended wait times at MPI barriers; and 2) inefficiencies in the construction of time stamps consumed inordinate computational resources during spike delivery. Feature 1 appears primarily for smaller simulations, while feature 2 is only apparent at the current limit of neural networks on the largest supercomputing and can only be identified through the use of profiling in light of clear computing models. By improving the underlying code, NEST performance has been significantly improved (on the order of 25% for each feature) and we have improved weak-scaling for simulations at HPC scales.

nest:: Neural Simulation Tool

- NEST [1, 2] is developed by the “NEST Initiative”, an international non-profit organization
- Scales from notebooks to super-computers
- Implemented in C++ using hybrid parallelism (MPI+OpenMP)
- Simulations are programmable in Python and SLI

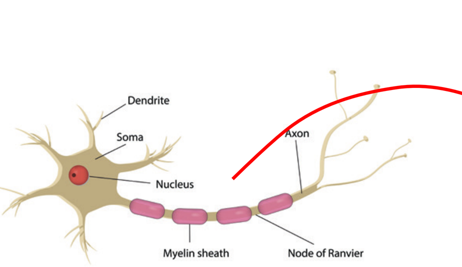
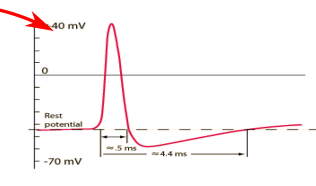
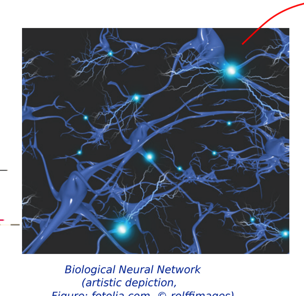
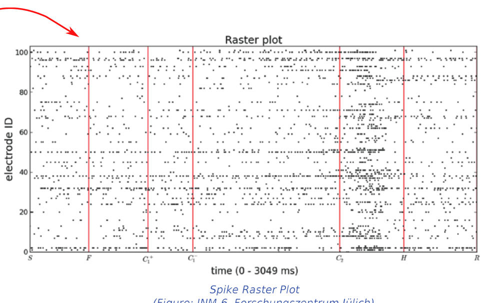


Figure: fotolia.com, © joshya



Spike


Biological Neural Network (artistic depiction; Figure: fotolia.com, © roffimages)



Spike Raster Plot (Figure: INM-6, Forschungszentrum Jülich)

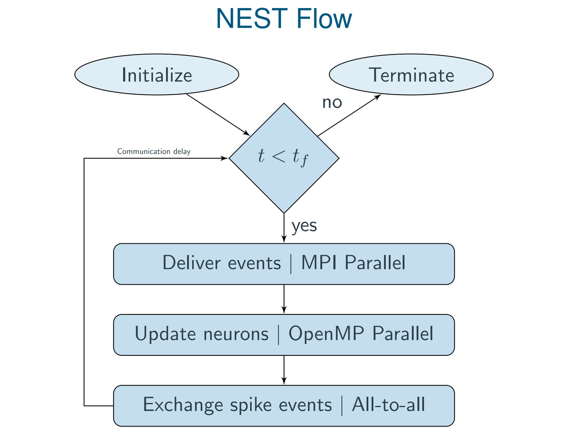
Understanding performance

The scale of biological problems that can be investigated via a simulator is constrained by the performance of the software on the available hardware. Performance enhancement leads to the capability to ask new scientific questions. To improve software in an evidence-directed way, several approaches are available:

”Real” Simulation	Full simulations can be instrumented with tools such as HPC Toolkit and SCORE-P to collect live data. Resource intensive
Dry-run mode	NEST can be run on a single rank with simulated communications to predict the results of a full simulation
Theoretical Model	A theoretical model of NEST has been developed to predict the impact of changes to code, and to interpret the sources of performance seen from empirical measurements [3]

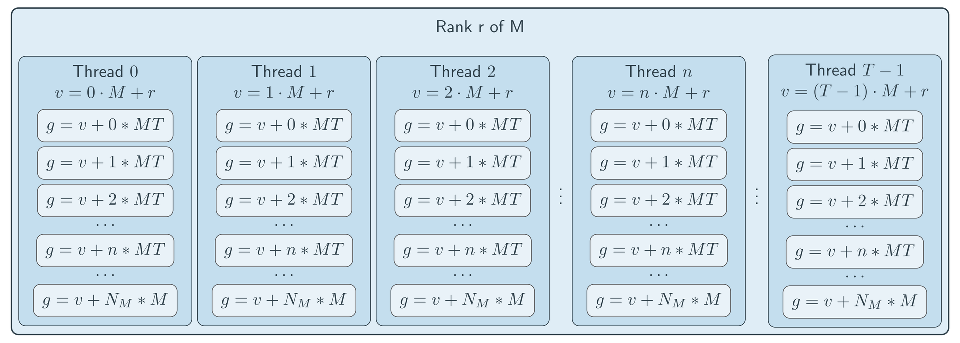
Structure of a NEST simulation

NEST Flow



Measured and modeled data on this poster refer to the the post-initialization stages of the simulation. Initialization can include a significant “network construction phase”.

Data distribution



NEST is hybrid parallel: each ‘node’ in a cluster runs one MPI rank which is subdivided into OpenMP threads. Spikes that occur within a time window are collected from all ranks and send to all other ranks, and then delivered to the neurons which simulate their responses. Neurons are grouped in virtual processes (threads) which act in parallel.

Theoretical model of simulation stage

Variables:

M	Number of MPI processes
T	Number of threads / process
N_M	Memory fill factor (neurons per process)
N	Total number of neurons
F	Spike frequency / neuron
F_{STDP}	Facilitation frequency / STDP synapse
K	Total number of incoming synapses / neuron
K_{STDP}	Number of incoming STDP synapses / neuron

Relations:

$$p_{rel} = 1 - \exp(-K/MT)$$
$$N = N_M \cdot M \cdot 11250$$
$$K = K_{scale} \cdot 11250$$

Free parameters:
Empirical fitting: $W_0 \ldots W_8$

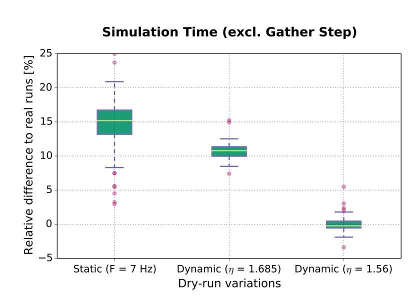
Estimated Time:

$$\begin{aligned} \hat{t} = \; & W_0 \cdot N/MT && \text{Neuron update} \\ & + W_1 \cdot F_{STDP} \cdot K_{STDP} \cdot N/MT && \text{Synapse update} \\ & + W_2 \cdot MT && \text{Buffer Jumps} \\ & + W_4 \cdot MT && \text{Main delivery loop} \\ & + W_6 \cdot FNT && \text{Thread collisions} \\ & + W_7 \cdot FN && \text{All spikes} \\ & + W_8 \cdot p_{rel} \cdot FN && \text{Relevant spikes} \end{aligned}$$

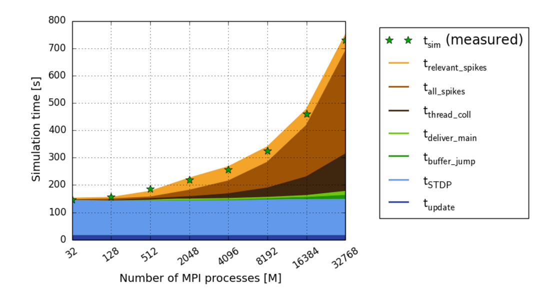
Updated terms for next generation NEST:

$$\begin{aligned} & W_2 \cdot W_3 \cdot M && \text{Buffer Jumps (from } MT\text{)} \\ & W_4 \cdot W_5 \cdot M && \text{Main loop overhead (from } MT\text{)} \end{aligned}$$

Dry-run model



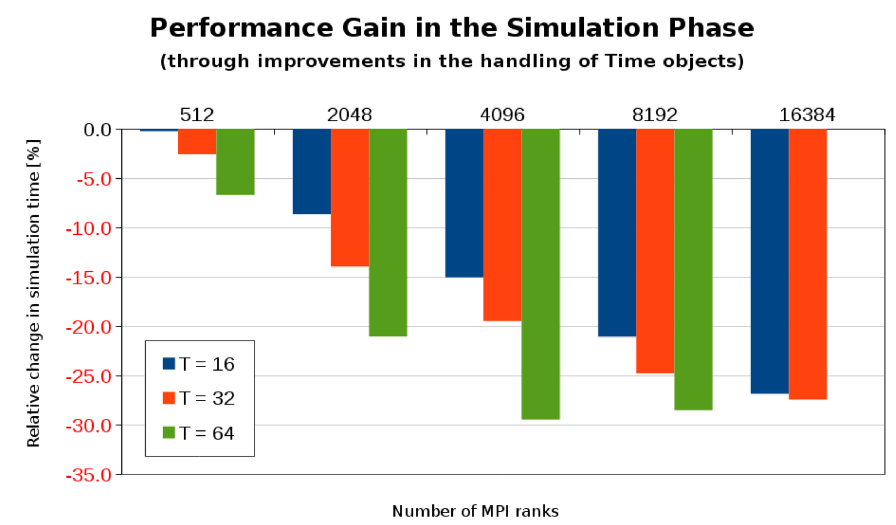
For dry-run mode, NEST is run on a single node with global communications either replaced with a fixed spike rate F or by a dynamic firing rate determined by η, which matches well with full simulations (left). Initially, poor weak scaling was measured for spike and thread collection terms (right).



Premature optimization

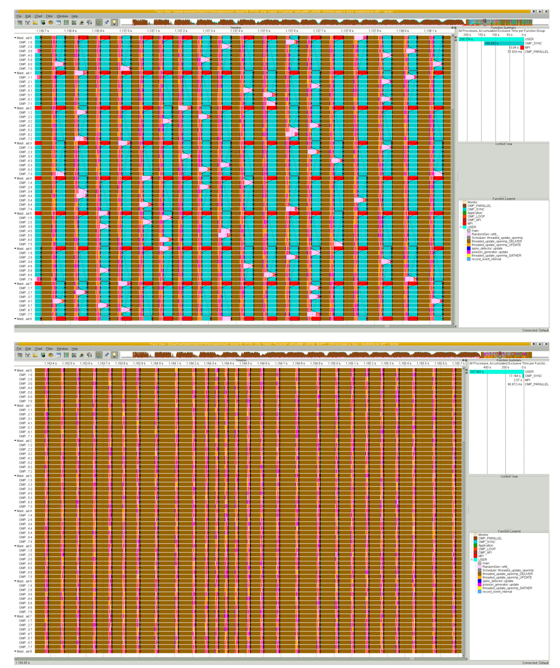
Time stamps	Time objects cached various formats (integer steps, integer tics & floating point time...) which in practice reduced performance on the order of 25%.
Buffering random numbers	for each rank lead to intermittent load imbalance which grows as a function of network size. Random sequences would be initially buffered for use on demands, and when this entropy pool was depleted, it would be refilled on a per-rank basis.

Time stamp change



The time stamp object was originally composed of an integer ‘tics’ (minimum time unit), a double floating point time in milliseconds, and the number of update steps of the time unit. The class was simplified to only contain tics, and at the central dispatch loop, the tics are now passed without a class wrapper.

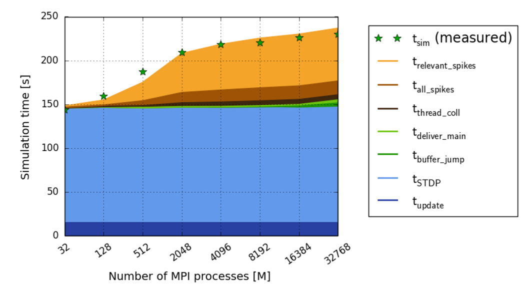
Random number buffering



A simulation of a reduced visual cortex model [4] was benchmarked with SCORE-P [5] and visualized with Vampir [6]. The simulation was composed of 16 areas with 80000 neurons each distributed over 32 compute nodes on JUQUEEN [7] (256 VPs). Brown: spike routing, cyan: `OMP_SYNC`, red: MPI communications.

The light pink is random number buffer refills which produce large wait times for non-refilling threads. By turning off buffering and producing random numbers on demand, this load imbalance is eliminated (bottom Vampir image), reducing runtime of the simulation stage from 216 s to 160 s. For smaller simulations with less spike routing, the effective reduction can be larger.

Scaling



Weak scaling improvements for Balanced Brunel network after current improvements and potential future improvements for next generation NEST (left). A combination of software implementation improvements shown here developed through HPC debugging tools, and further software architectures improvements through performance modeling has lead to improvements seen at all scales, but particularly at the larger scales currently becoming common in neuroscience.

Acknowledgments & bibliography

This work was funded by the Helmholtz Association through the Portfolio Theme “Supercomputing and Modeling for the Human Brain”.

[1] Markus Diesmann and Marc-Oliver Gewaltig. NEST: An environment for neural systems simulations. 58:43–70, 2001.

[2] NEST Initiative. NEST (Neural Simulation Tool). Software, 2014. URL `https://github.com/nest/nest-simulator`.

[3] W. Schenck, A. Adinets, Y. Zaytsev, D. Pleiter, and A. Morrison. Performance model for large-scale neural simulations with nest. In *Supercomputing 2014, SC14, New Orleans, USA*, 16 Nov 2014 - 21 Nov 2014. Extended Poster Abstract.

[4] Maximilian Schmidt, Sacha van Albada, Rembrandt Bakker, and Markus Diesmann. Integrating multi-scale data for a network model of macaque visual cortex. *BMC Neuroscience*, 14(Suppl 1):P111–P111, July 2013. ISSN 1471-2202. URL `http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3704432/`. Abstract.

[5] Andreas Knüpfer, Christian Rössel, Dieteran Mey, Scott Biersdorff, Kai Diethelm, Dominic Eschweiler, Markus Geimer, Michael Gerndt, Daniel Lorenz, Allen Malony, WolfgangE. Nagel, Yury Oleynik, Peter Philippen, Pavel Saviankou, Dirk Schmidl, Sameer Shende, Ronny Tschüter, Michael Wagner, Bert Wesarg, and Felix Wolf. Score-p: A joint performance measurement run-time infrastructure for periscope, scalasca, tau, and vampir. In Holger Brunst, Matthias S. Müller, Wolfgang E. Nagel, and Michael M. Resch, editors, *Tools for High Performance Computing 2011*, pages 79–91. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-31475-9. doi: 10.1007/978-3-642-31476-6_7. URL `http://dx.doi.org/10.1007/978-3-642-31476-6_7`.

[6] Andreas Knüpfer, Holger Brunst, Jens Doleschal, Matthias Jurenz, Matthias Lieber, Holger Mickler, MatthiasS. Müller, and WolfgangE. Nagel. The vampir performance analysis tool-set. In Michael Resch, Rainer Keller, Valentin Himmler, Bettina Krammer, and Alexander Schulz, editors, *Tools for High Performance Computing*, pages 139–155. Springer Berlin Heidelberg, 2008. ISBN 978-3-540-68561-6. doi: 10.1007/978-3-540-68564-7_9. URL `http://dx.doi.org/10.1007/978-3-540-68564-7_9`.

[7] Dirk Brömmel, Estela Suarez, Boris Orth, Stephan Graf, Ulrich Detert, Dirk Pleiter, Michael Stephan, and Thomas Lippert. Paving the road towards pre-exascale supercomputing. In *NIC Symposium 2014*, number FZJ-2014-01327. Jülich Supercomputing Center, 2014.