A Comparative Evaluation of Xeon Phi Platforms Based on a Hodgkin-Huxley Neuron Simulator

George Chatzikonstantis, Diego Jiménez, Esteban Meneses, Christos Strydis, Harry Sidiropoulos, Dimitrios Soudris

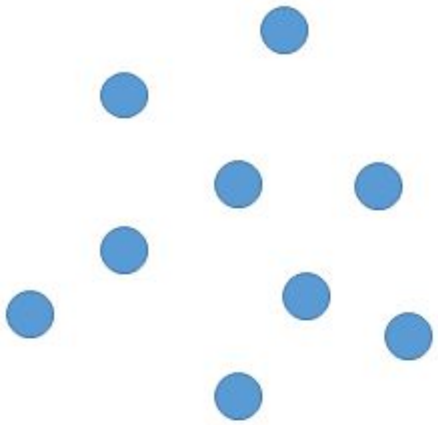
-

Microprocessors and Digital Systems Lab, National Technical University of Athens
Advanced Computing Laboratory, Costa Rica National High Technology Center
School of Computing, Costa Rica Institute of Technology
Neuroscience Department, Erasmus Medical Center Rotterdam

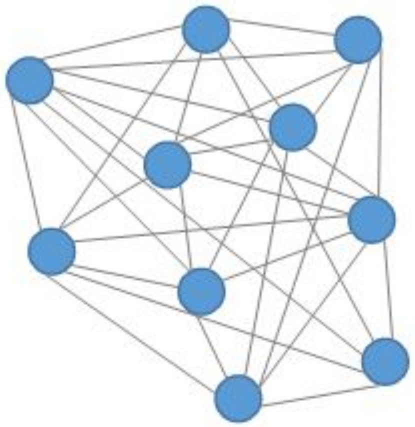
Source: Dr. Greg Dunn
nd Dr. Brian Edwards - *Self*
eflected

Problem Complexity

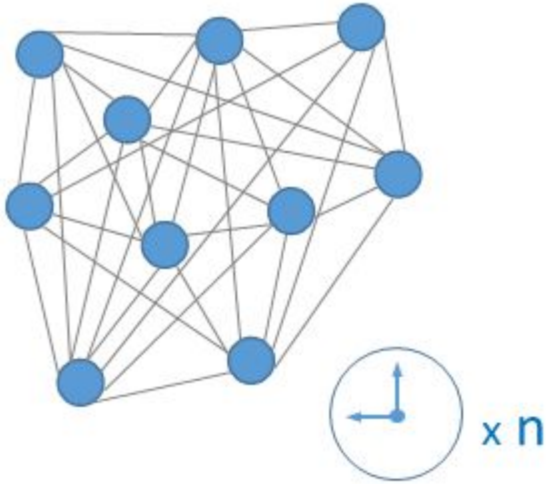
Many FLOPs per neuron



Massive network

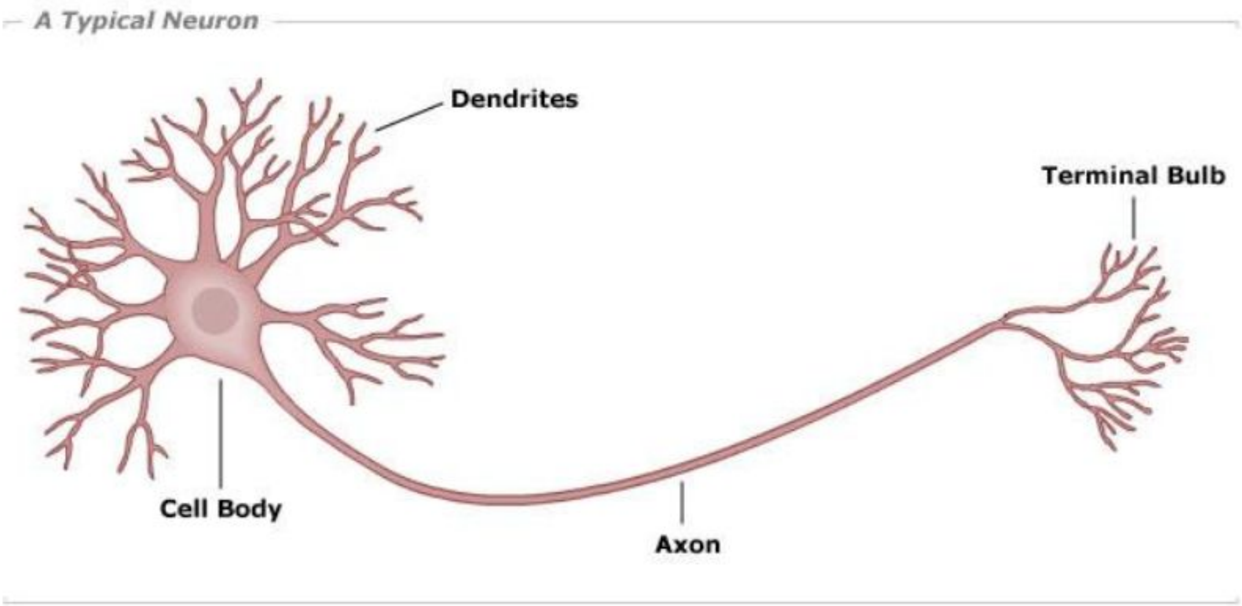


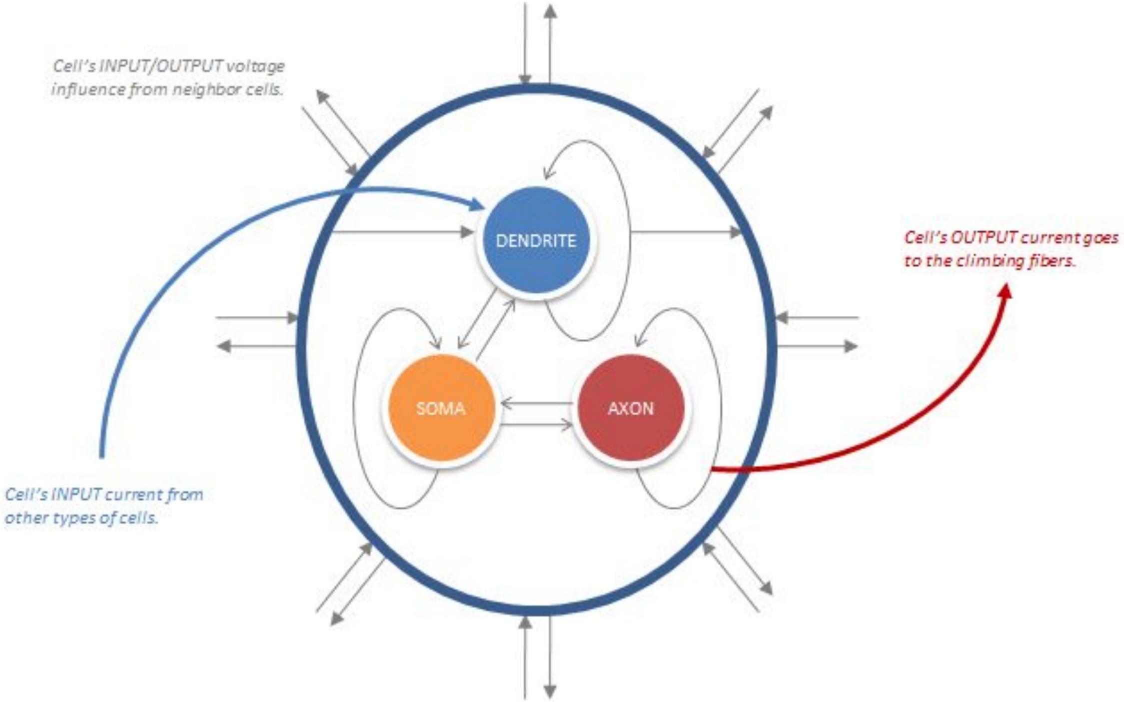
Densely connected networks



Real-time response is currently impossible

Infoli Simulator - Description



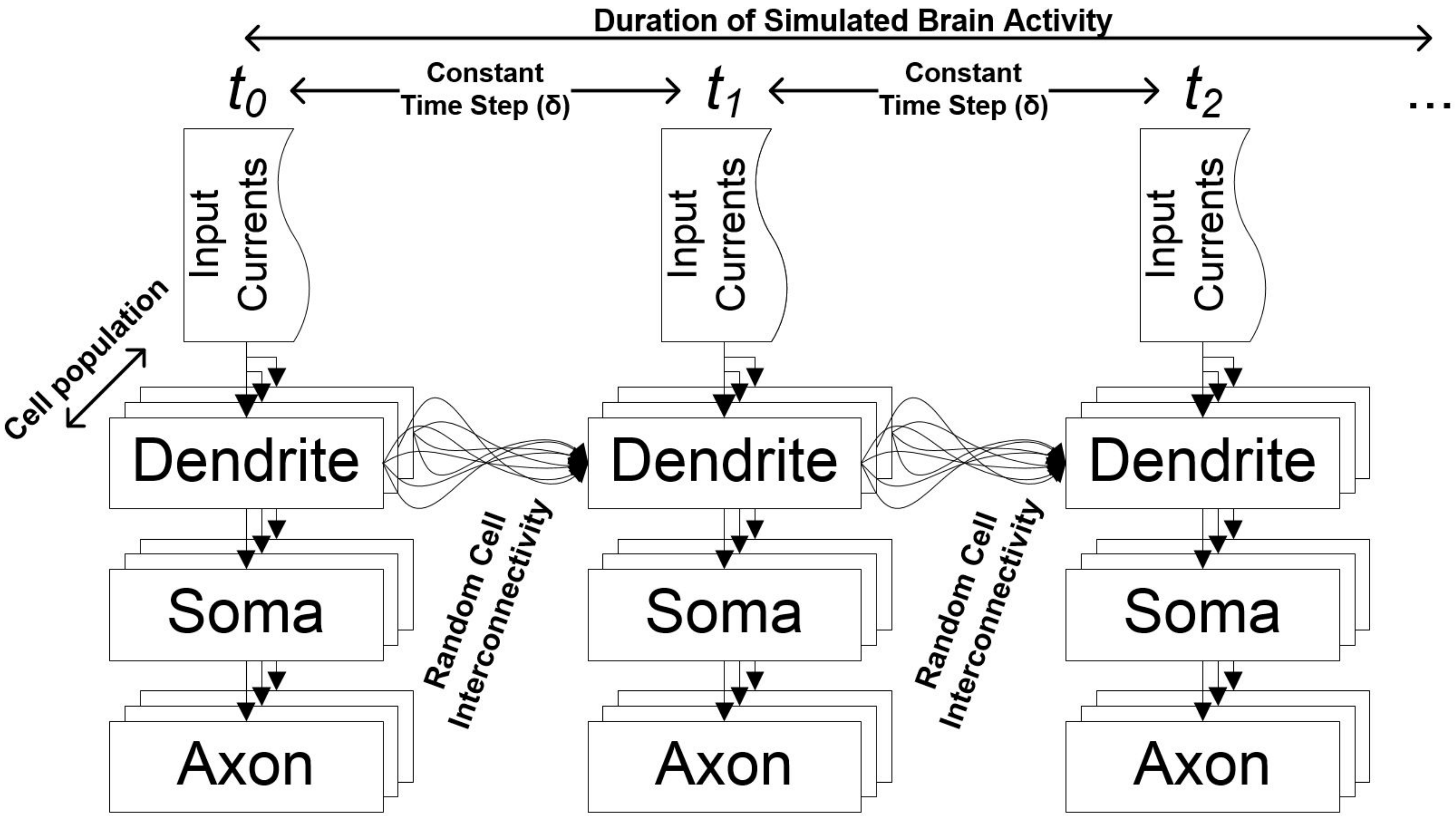


- Tri-compartmental model:
  Dendrite: communication
  Soma (body): computation
  Axon: output

- Gap Junction mechanic
  The communication between dendrites in the network

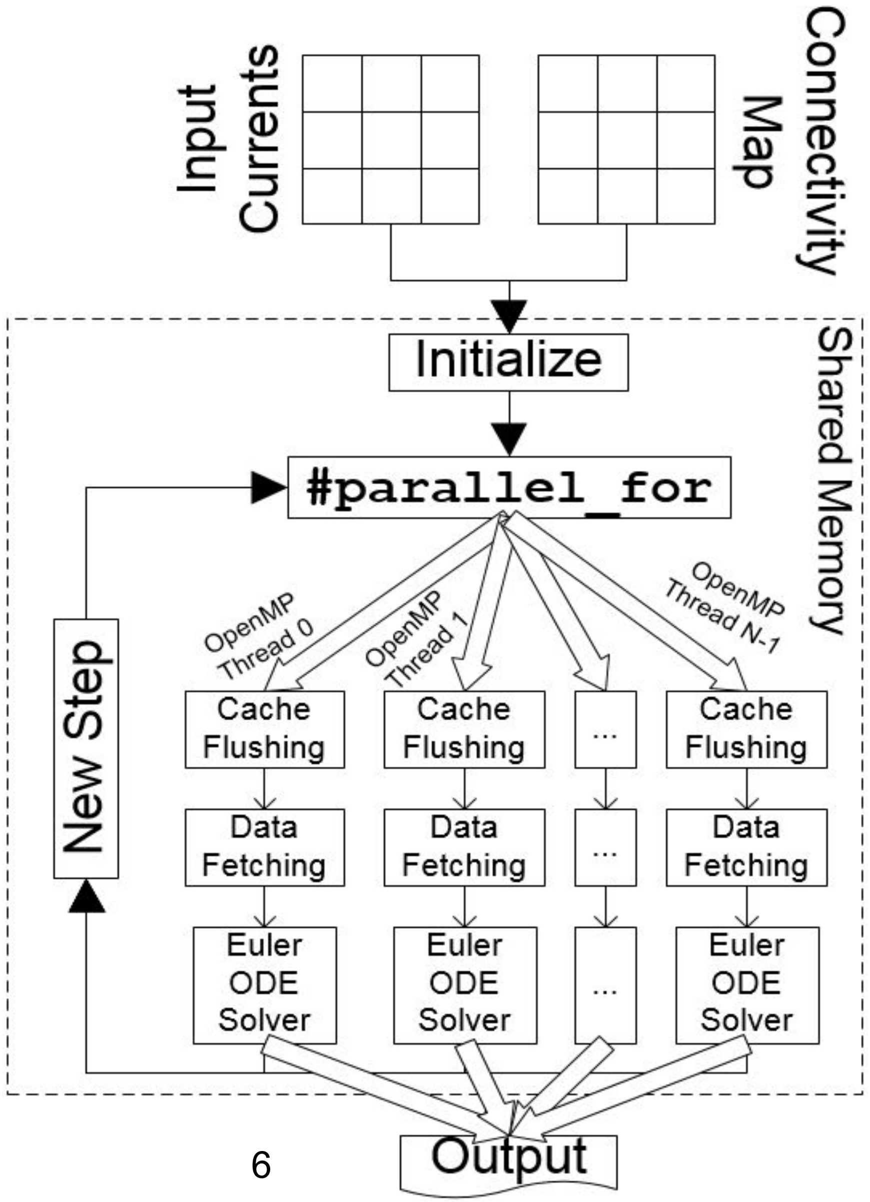
Performance Bottleneck!

Infoli Simulator - Description



Infoli Simulator - Parallelization

- *OpenMP threads, up to 240 on the KNC and 256 on the KNL*
- Data Partitioning:
  - ■Each thread handles a subnetwork
  - ■Network is divided as evenly as possible
- *Need for data exchange between threads*
- Neurons are calculated independently:
  - ■Threads operate in parallel
  - ■Each thread vectorizes calculation for more parallel neuron processing



From Knights Corner to Knights Landing

Intel's 1st Generation Xeon Phi: Knights Corner Coprocessor Card

Model: 3120p

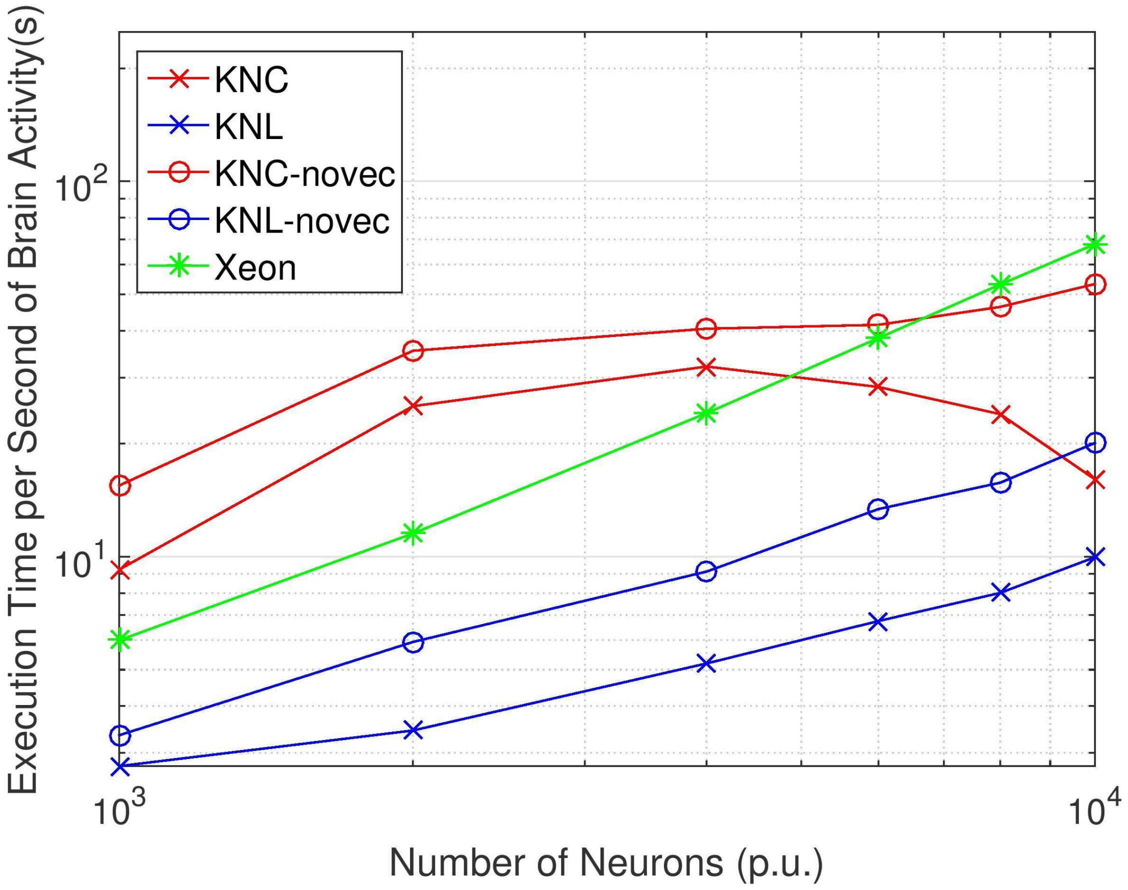
Intel's 2nd Generation Xeon Phi: Knights Landing Processor

Model: 7210

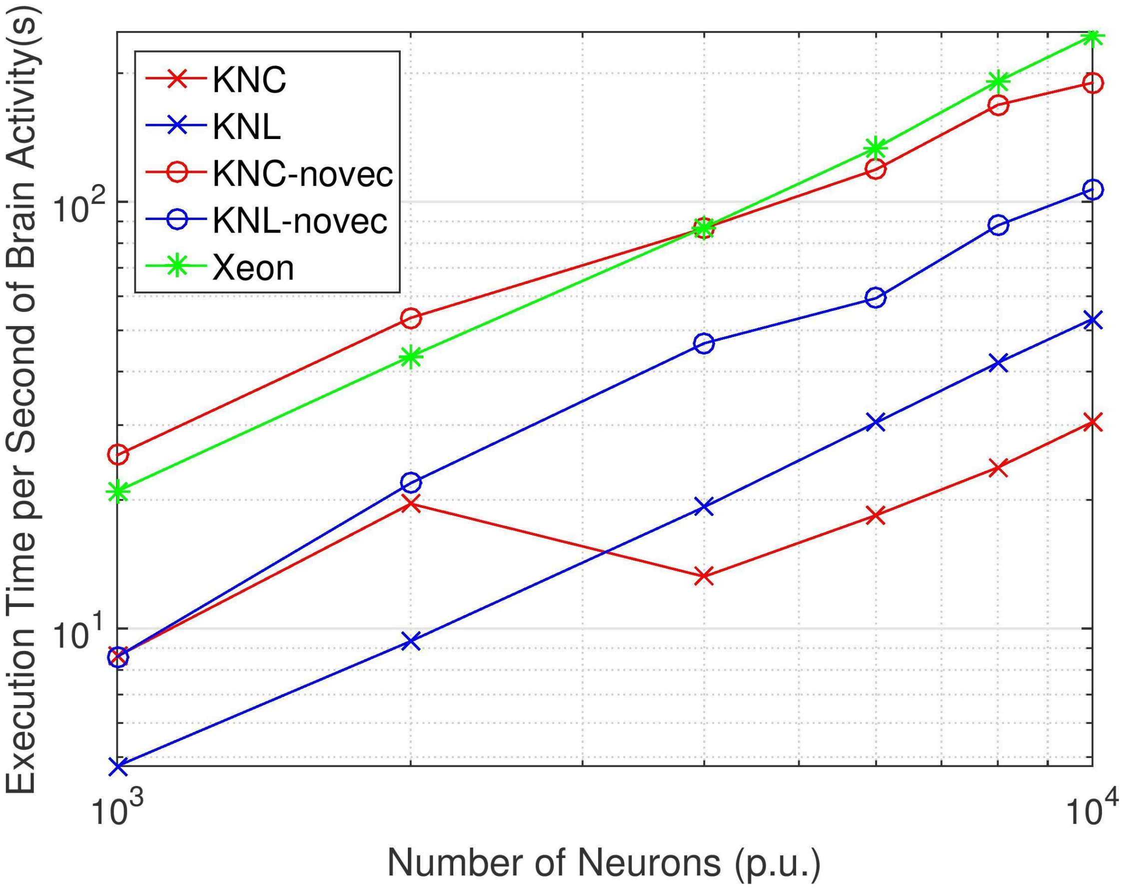
Xeon Baseline model: E5-2609-v2 (4 cores, Ivy Bridge)

- *Out-the-box measurements from the KNC on the KNL.*
- **Ease of transferring:** only recompilation needed
- KNL vs KNC?
  - Better single-threaded performance (3x TFPs)
  - More VPUs, better vectorization support
  - High Bandwidth MCDRAM (set to cache mode)
  - Increased amount of cores, maximum amount of threads
- Experimental evaluation
  - Small (1000) to large (10k) neuron networks
  - Connectivity densities: from 0 up to 1 k GJs per neuron.
  - Exploration of simulation speed, energy used and thread efficiency.

Results - Execution time

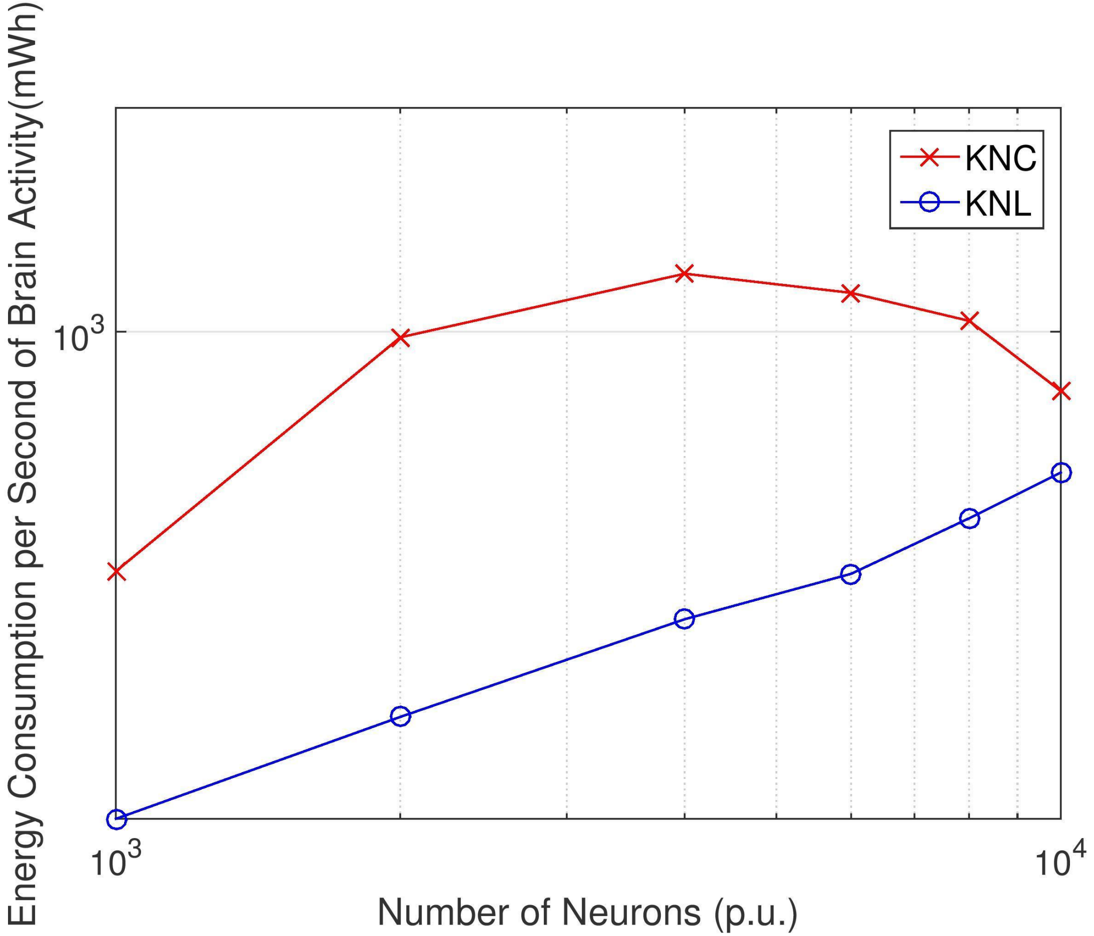


Low-density networks

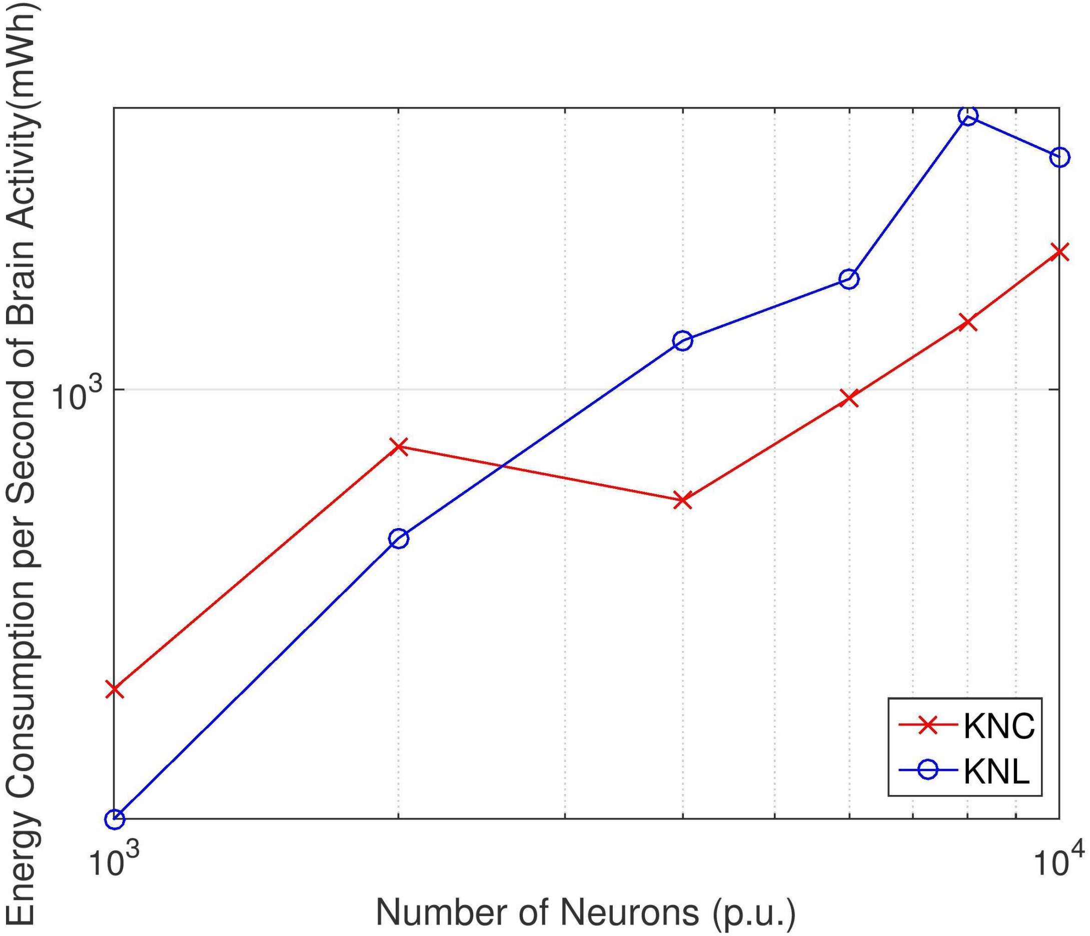


High-density networks

Results - Energy Consumption

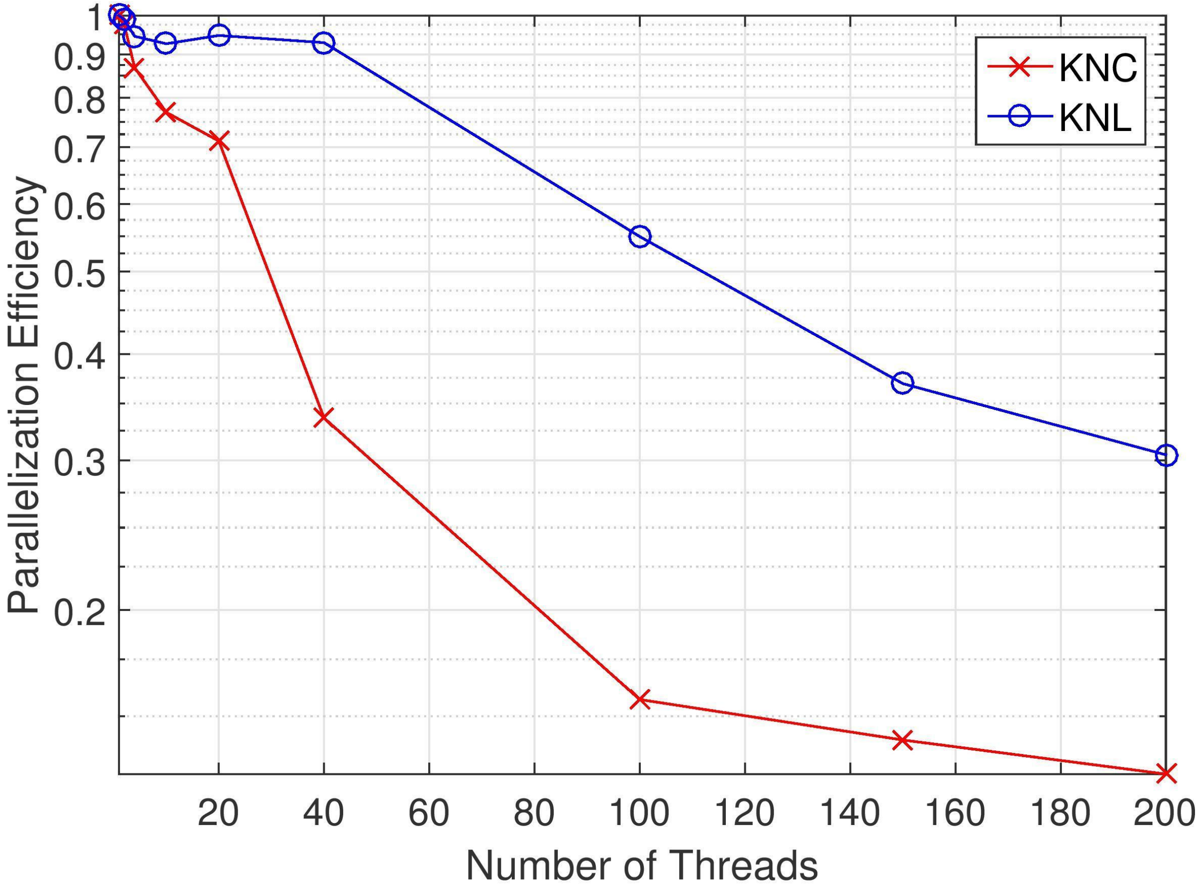


Low-density networks

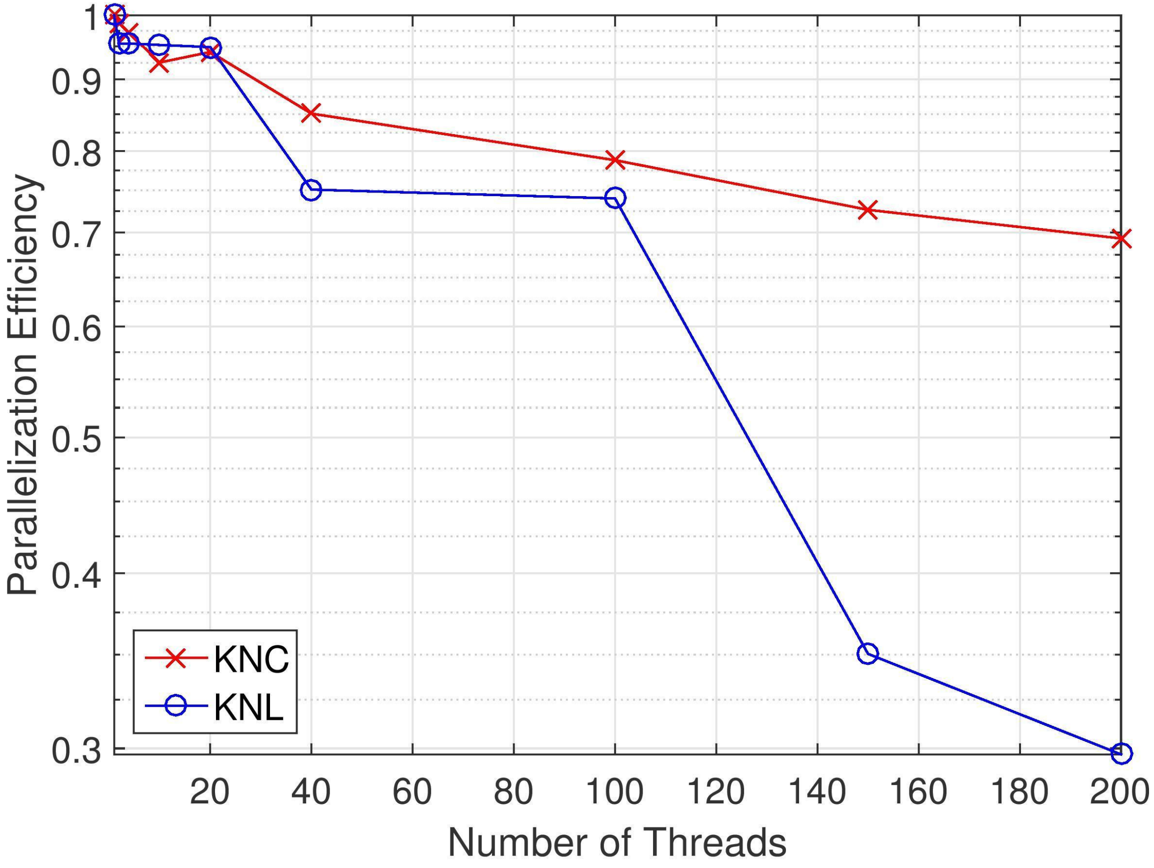


High-density networks

Results - Efficiency



High-density network of 1000 neurons



High-density network of 10k neurons

Results - Analysis

- Sparse networks are more serial in nature, so they operate well on KNL (superior single-threaded performance).
- Denser networks heavily favor vectorization-enabled implementations:
  - Vectorization on the KNC is significantly better after a certain point.
  - KNL performance is worse for some of the heaviest workloads.
- KNL´s lower TDP leads to significant energy gains.
  - Gap lessens with higher workload.
  - On heavier workloads, KNL´s lower TDP offset by increased simulation times.
- KNL very efficient for 1 thread per core, however efficiency takes a significant hit past 100 threads.
- KNC retains acceptable efficiency for 200 threads.

Conclusions and Insights

- On average, 2.4x speedup, comparable to expected single thread performance upgrade of KNL over KNC (3x).
- Lower TDP leads to overall energy savings (~50%) on KNL. Up to 75% saving on low density networks!
- Thread efficiency suffers on the KNL possibly because of lack of fine-tuning of the application to the architectural details of the platform.
    - Best practice suggests ~2 threads per KNL core.
- KNL displays greater predictability in performance.

Future Work

- Fine tuning for the KNL:
  - VPU optimal usage
  - Thread efficiency

- Exploration of MCDRAM modes and clustering modes

- Hybrid MPI + OpenMP for multinode systems
  - Usage of Intel’s Omnipath technology

Thank you!