WP5: Neural Net Distributed Simulation (DPSNN).

Energy to solution and speed of «embedded» vs «server» dual-socket quad-core nodes.

ARM Cortex-A15 in 2x NVIDIA Jetson TK1 boards vs Intel Xeon quad-core in dual-socket node

Pier Stanislao Paolucci
for INFN – APE Lab:

R. Ammendola, A. Biagioni, O. Frezza, F. Lo Cicero, A. Lonardo,
M. Martinelli, P. S Paolucci, E. Pastorelli, F. Simula, P. Vicini

DPSNN – STDP Status

- Distributed Simulator of Plastic Spiking Neural Networks with synaptic Spike Timing Dependent Plasticity
- State-of-the-art Application AND architectural benchmark:
- Designed in EURETILE EU Project 2011-2014: arXiv:1310.8478
- Speed-up of CORTICONIC EU Project simulations (FIRST PART of this presentation):
  - x420 speed-up on 4 servers and 100 MPI processes
- **COSA: comparison of power/energy of distributed execution on embedded and server quad-cores:** (SECOND PART of this presentation)
  - **arXiv:1505.03015**
- Benchmark in next-to-start EXANEST EU Project
- …and in piCOLO proposal …

Acceleration (up to now, x 420) of CORTICONIC simulations through parallelization/distribution techniques on many-processor architecture

Elena Pastorelli and Pier Stanislao Paolucci for

INFN Roma - APE Lab*

P. Del Giudice, M. Mattia - ISS Roma

*INFN Roma – APE Lab: R. Ammendola, A. Biagioni, O. Frezza, F. Lo Cicero, A. Lonardo, M. Martinelli, P. S Paolucci, E. Pastorelli, F. Simula, P. Vicini

CORTICONIC – 2015 04 28 - Barcellona

A Challenging Problem

- **The simulation of the cortical field activity can be accelerated using parallel/distributed many-processor computers. However, there are several challenges, including:**
  - Neural networks heavily interconnected at multiple distances, local activity rapidly produces effects at all distances → Prototype of non-trivial parallelization problem
  - Each neural spike originates a cascade of synaptic events at multiple times: $t + \Delta t_s$ → Complex data structures and synchronization. Mixed time-driven (delivery of spiking messages and neural dynamic) and event-driven (synaptic activity)
  - Multiple time-scales (neural, synaptic, long and short term plasticity models) → Non-trivial synchronization at all scales
  - Gigantic synaptic data-base. A key issue for large scale simulations → Clever parallel resource management required.

x 420 acceleration delivered

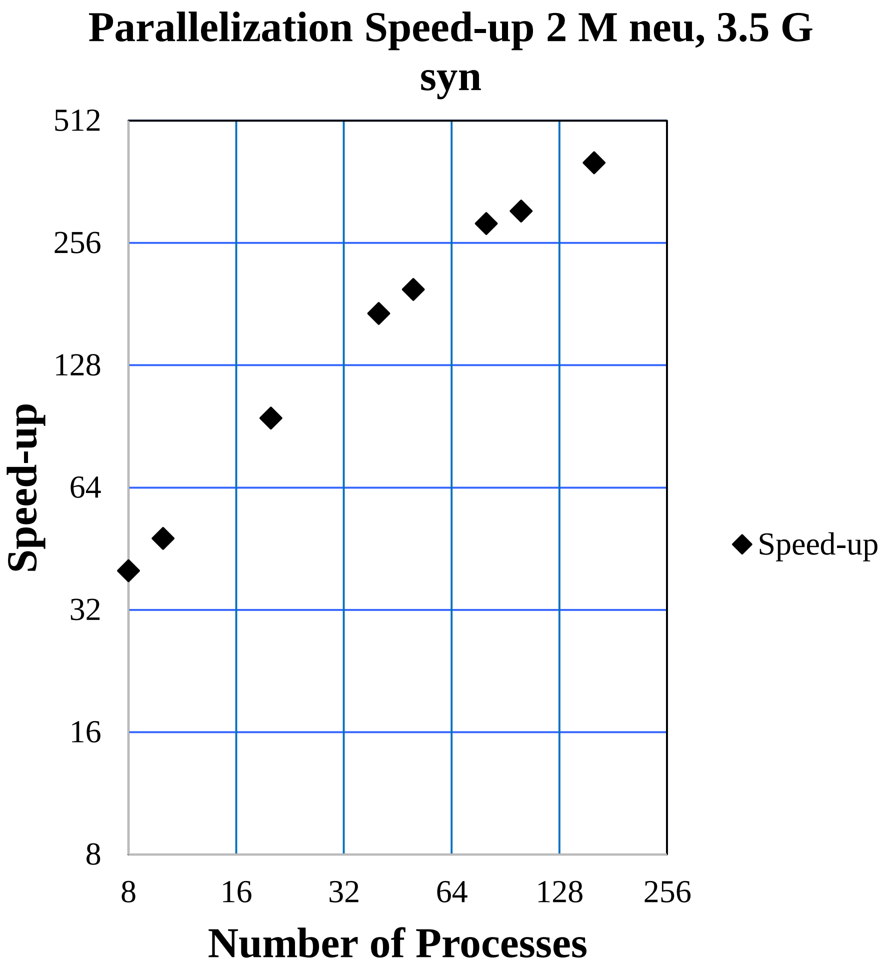
- 60 sec simulated activity of 2 M neuron, 3.5 G synapses
  - Organized in a grid of 40x40 cortical columns
- **Starting point (before parallelization)**

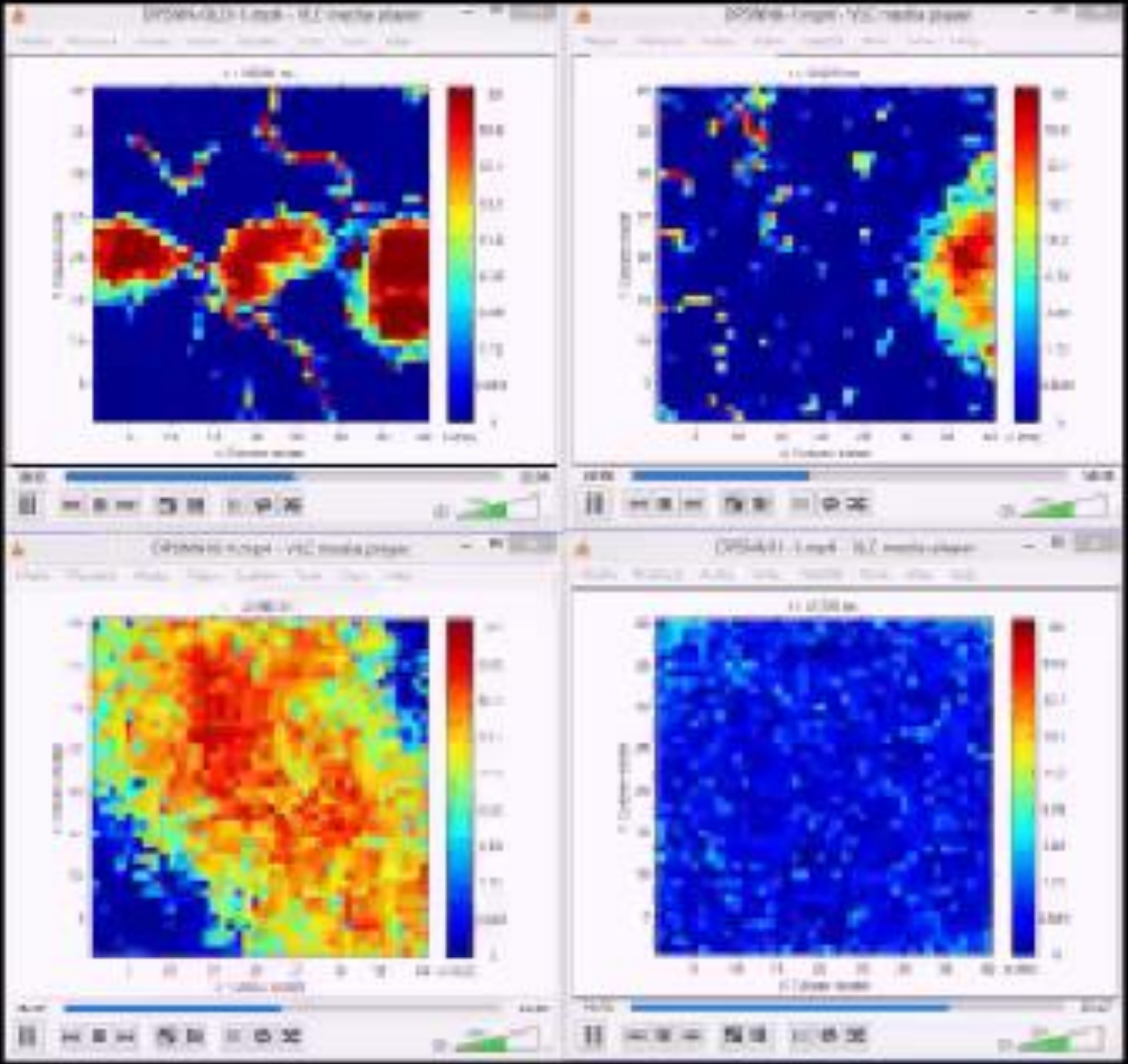
→ **1000^{h} elapsed time** required (single process running on a single processor)

- **After application of our distribution/parallelization techniques**
- **→ now requires ~2^{h} 20^{m} elapsed time**
  - o **x 420 acceleration on 4 servers** on 100 processes
    - o (dual socket Intel® Xeon® CPU E5-2650 0 @ 2.00GHz – 8 hw core per socket)
  - o 6.1 G syn fit on a 4 server cluster memory
  - o for an introduction to our mixed time and event driven parallelization/distribution techniques, see arXiv:1310.8478

CORTICONIC Speed-up through parallelization

- **Strong scaling (i.e. fixed problem size) of the 2M neuron, 3.5 G synapses configuration measured**
- **For larger problem sizes, good scaling expected using higher number of hardware resources and software processes**
- Exec. time of original non parallel code used as speed-up reference



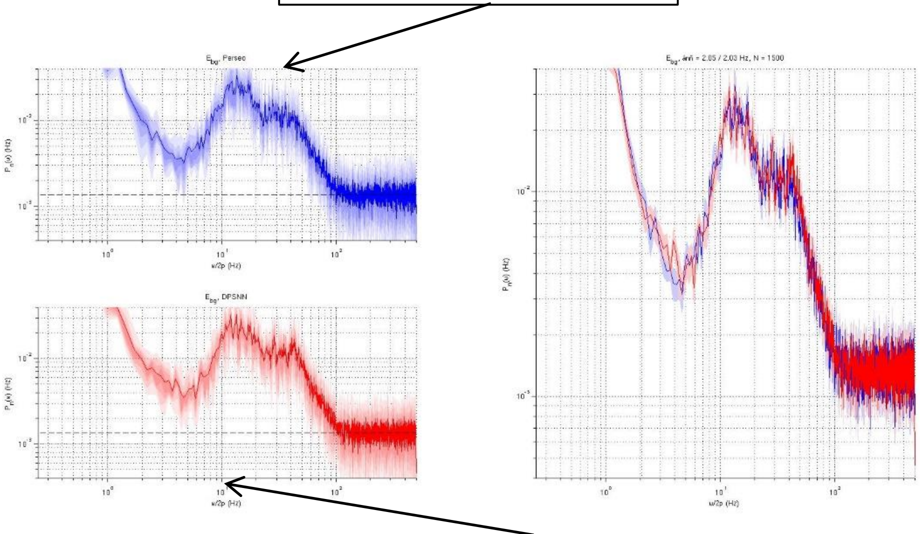

VLC media player
VLC media player
VLC media player
VLC media player

Validation

The equivalence between the spiking activity simulated by the original (scalar) and the accelerated (parallel) code has been verified at all hierarchical levels:

- Single neuron
- Poissonian stimulus
- Synaptic messaging
- Sub-population
- Cortical Module
- Cortical Field

Original (non parallel) code

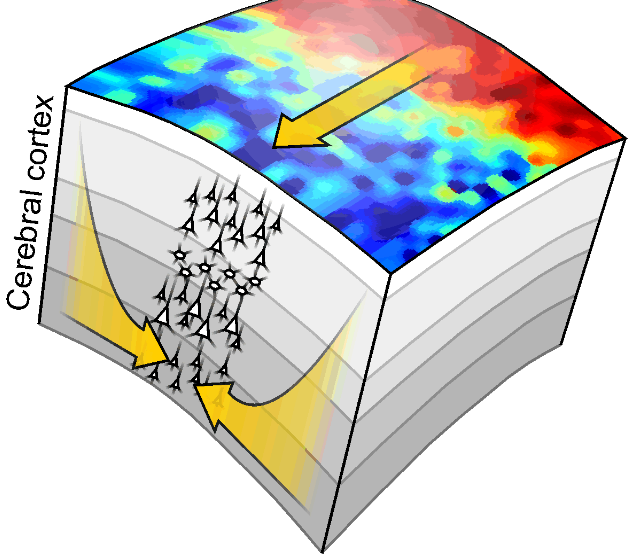


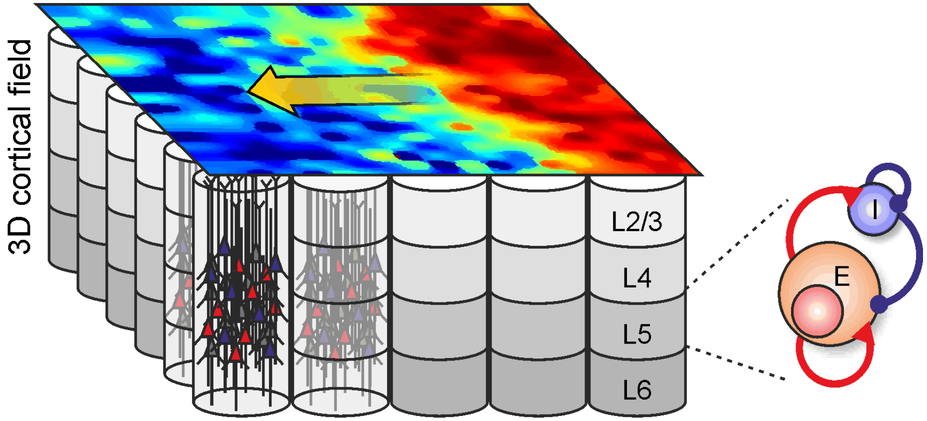
Parallelized code

Main tool: analysis of power spectrum at different hierarchical levels

Conclusion/Future

- New parallel/distributed simulator of spiking neural networks relying on mixed time- and event-driven integration
- Accelerated CORTICONIC simulations (up to now, x 420), Reduced mem consumption → large networks possible
- Future developments:





COSA: Jetson boards set-up details

- Initial standalone bootstrap of each board (no network)
- Ubuntu 14.04 LTS (GNU /Linux 3.10.24-g6a2d13a arm7v1)
- sudo apt-mark hold xserver-xorg-core
- cd {$HOME}/NVIDIA-INSTALLER; sudo ./installer.sh; reboot now
- Assign name to each board (e.g. tk00.ape and tk01.ape)
  (and modified /etc/hostname and /etc/hosts on each board)
- Use a Ethernet switch to interconnect the Jetson boards
- Install c++: sudo apt-get install g++
- Install OpenMPI 1.8.1 on each Jetson board
  - mkdir /opt/openmpi-1.8.1; cd ~; wget https://www.open-mpi.org/...
  - tar –xvf openmpi-1.8.1.tar.gz; cd openmpi-1.8.1; mkdir build; cd build
  - ../configure -- prefix=/opt/openmpi-1.8.1; make; sudo make install
  - Added to ~/.bashrc (just at the beginning, before the exit on non interactive login):
    - export PATH="$PATH:/opt/openmpi-1.8.1/bin"
    - export LD_LIBRARY_PATH="$LD_LIBRARY_PATH:/opt/openmpi-1.8.1/bin"
- mpirun –np 8 – host tk00.ape, tk01.ape hostname
  - Prints 8 output lines
    - tk00
    - tk01
    - …
    - tk01

COSA: same neural net sim. launched on

- “embedded platform” “node”:
  - dual-socket node emulator:
    - 2 x NVIDIA Jetson TK1 board (Tegra K1 chip):
    - 2 x quad-core ARM Cortex-A15@2.3GHz,
    - Ethernet interconnected (100 Mb mini-switch)
    - DPSNN distributed on 8 MPI processes / node
- “server platform” node:
  - dual-socket server SuperMicro X8DTG-D 1U:
    - 2 x quad-core Intel Xeon CPUs (E5620@2.4GHz)
    - 2 x HyperThreading
    - DPSNN distributed on 16 MPI processes / node

Energy to Solution, Speed and Power

- 2.2 micro-Joule per simulated synaptic event on the “embedded dual socket node”
  - 4.4 times better than spent by “server platform”
- instantaneous power consumption: “embedded ”14.4 times better than “server”
- “server” platform 3.3 faster than “embedded”
- All inclusive, measured using amperometric clamp on 220V@50Hz power supply on:
- Details in arXiv:1505.03015 – May 2015

DPSNN in COSA: possible next steps

- Accelerate tasks on «embedded» nodes, e.g.:
  - Random numbers
  - Interconnect optimizations
- Evaluate scaling, energy to solution, instantaneous power and speed on larger number of COSA «embedded nodes»
  - N x 2 x Tegra Boards (and from K1 to X1)
  - ARM + FPGA + custom interconnet
- Compare scaling with large «server» platforms
- Key point: low memory / embedded board ?