Advances in Neuromorphic Ecosystems

Alexander Dimitrov
Department of Mathematics and Statistics
WASHINGTON STATE UNIVERSITY VANCOUVER

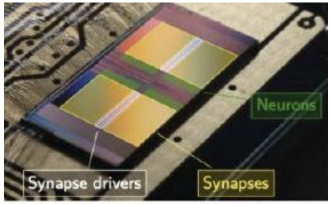
Lecture at the Third Summer School of the N3BG Group
“Neuroinformatics, Neural networks and Neurocomputers”,
30 April 2025, TU-Sofia, Bulgaria

System selection

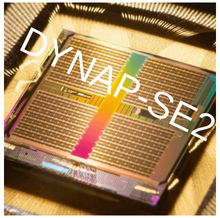
- Neuromorphic systems are relatively new and experimental.
- I tried to select neuromorphic systems that
  - Have relatively stable and evolving hardware
  - Have reasonable software support
  - Can scale (so some edge devices, but not many)
  - Are ‘interesting’ (usually event-based, so no GPUs or DNN accelerators).

Current neuromorphic systems

- Analog
  - BrainScaleS-2



  - DYNAPs (SynSense)



  - SynSense transitioned to digital architecture: Xylo and Speck

- Digital
  - SpiNNaker2 (TU Dresden/ SpiNNCloud)

  - Loihi (Intel)

  - Akida (BrainChip)

Good Source:
https://open-neuromorphic.org/

Why not these?

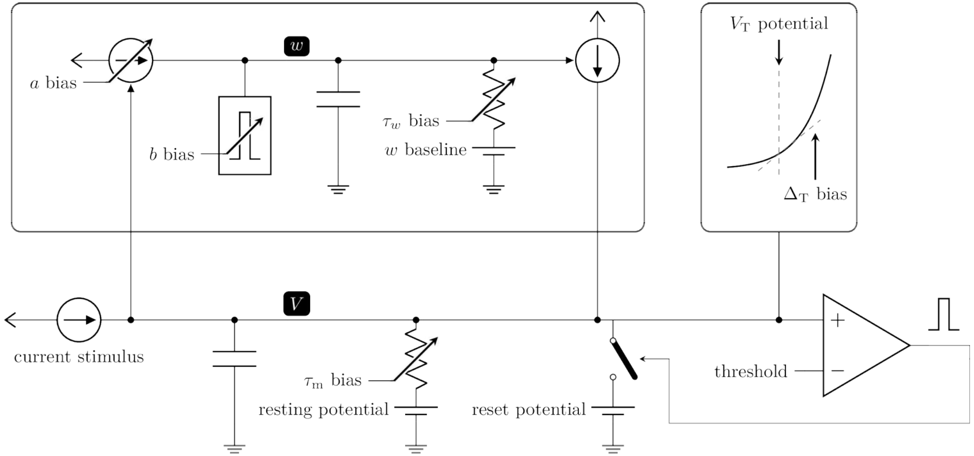
- T1 (Innatera), ReckOn, Odin (Frenkel), Darwin3
  - Too new and/or research chips. Less development support.
- NeuroGrid, ROLLS-INI, etc.
  - Defunct. Anyway, mainly only the developer could program those so not much application impact.
- Nvidia, Google, Intel, Graphcore, etc. ‘IPU’s
  - Mostly for DNNs. MAC & f()
  - Graphcore seems capable of SNN simulations.
- NorthPole (IBM)
  - TrueNorth was spiking and had dynamics; this is now just a low-precision inference engine. Neuromorphic in local memory and a *massive!* NoC.
- Tianjic
  - Spiking, but no on-chip learning

What do these systems have in common?

- Neuron models are stateful, have intrinsic dynamics.
- On-chip learning supported (weight dynamics).
- Neurons communicate with events, sparsely in time.
- (and the usual) many cores, local memory, scalable NoC.

Analog-hybrid systems

- Neuron and synaptic dynamics implemented in analog electronics, NoC digital. Uses the natural physics of the elements.
  - E.g. BSC-2:
    - AdExp neuron model
    - Exp synapse model
    - nonlinear dynamics

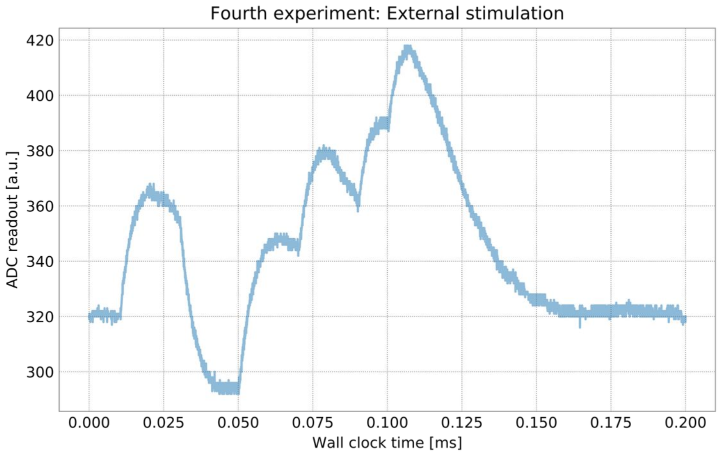


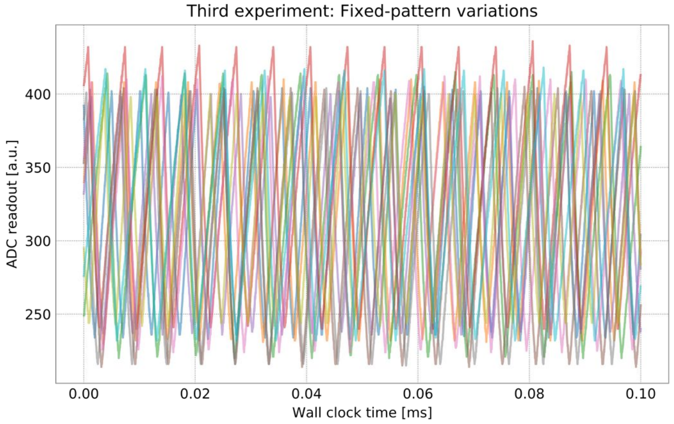
- Can be VERY fast and VERY energy-efficient.

Analog-hybrid systems

- Not for the faint-of-heart!
  - Unavoidable intrinsic noise. Fixed pattern variations. Dynamics depends on external conditions (e.g., T, E, B).
    - Like real neurons :)
  - Very robust algorithms needed!
  - From BSC2 site:

    … the mismatch of semiconductor fabrication results in inhomogeneous properties of the computational elements.
    … A default calibration is generated for every setup every night.





These are supposed to be the same neurons...

Software for analog

- PyNN works for BSC-2; generic AdExp SNNs. Common front end for digital neuromorpihc as well.
  - https://electronicvisions.github.io/documentation-brainscales2/latest/pynn-brainscales/index.html
  - https://wiki.ebrains.eu/bin/view/Collabs/neuromorphic/BrainScaleS/
  - https://www.ebrains.eu/modelling-simulation-and-computing/computing/neuromorphic-computing/
  - DYNAP-SE2 is geared more towards edge devices. Python interfaces like Rockpool.
  - https://rockpool.ai/index.html
- Many entry-level examples provided to try.

Digital neuromorphic

- Loihi and Akida are pure play SNN accelerators. SpiNNaker can also do that, but has more flexibility (different trade-offs). As noted, NorthPole is inference-only.
  - I see these as transition technology to Analog. But they can also solve many current complex problems.
- Main challenges:
  - Resource constraints (state size, parameter size, limited local memory)
  - Computing with stateful neurons
  - Computing with many small cores
    - Loihi’s Hala Point: 140K neurocores, 1G neurons, 128G synapses. 6U rack box, 2.5 kW …
    - SpiNNaker2: 10M ARM cores, ~1G neurons, 100G synapses. Room-size, 100kW or so
    - vs HPC: ~10^5 CPU cores, ~1K neurons/core, so 100M neurons.
with GPUs can get up to 1G neurons/100G synapses. Building-size, 20 MW

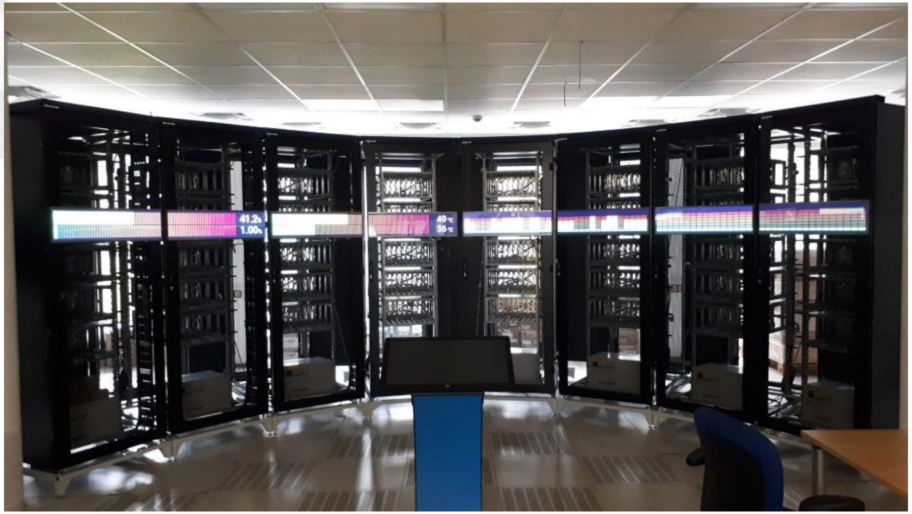
Comparative sizes for 1G neurons

Intel’s Hala Point

HPC

Human cortex ~ 16G neurons.

SpiNNaker 2



SpiNNaker 2

From Mayr’s presentation at FZJ

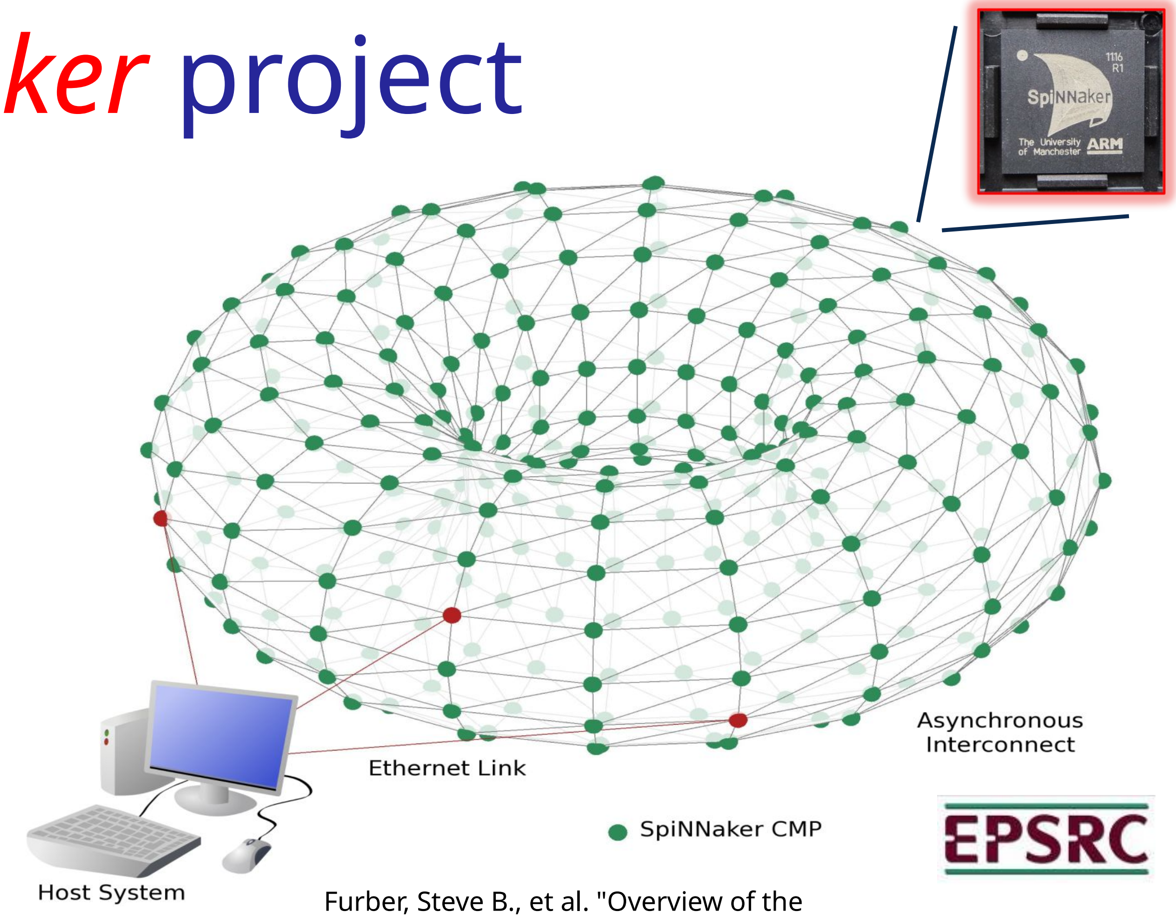
SpiNNaker 1

MANCHESTER 1824 The University of Manchester

SpiNNaker project

- Invented by Steve Furber, original ARM system architect
- A million mobile phone processors in one computer
- Strictly real-time architecture <1ms response time
- Able to model about 1% of the human brain…
- …or 10 mice!

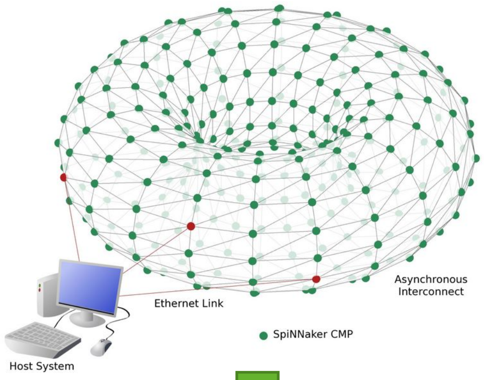
ARM®



EPSRC

Furber, Steve B., et al. "Overview of the spinnaker system architecture." *Computers, IEEE Transactions on* 62.12 (2013): 2454-2467.

SpiNNaker2



10 Mio ARM M4F cores in 22FDX, each with…

Dynamic Power Management
- DVFS and PSO

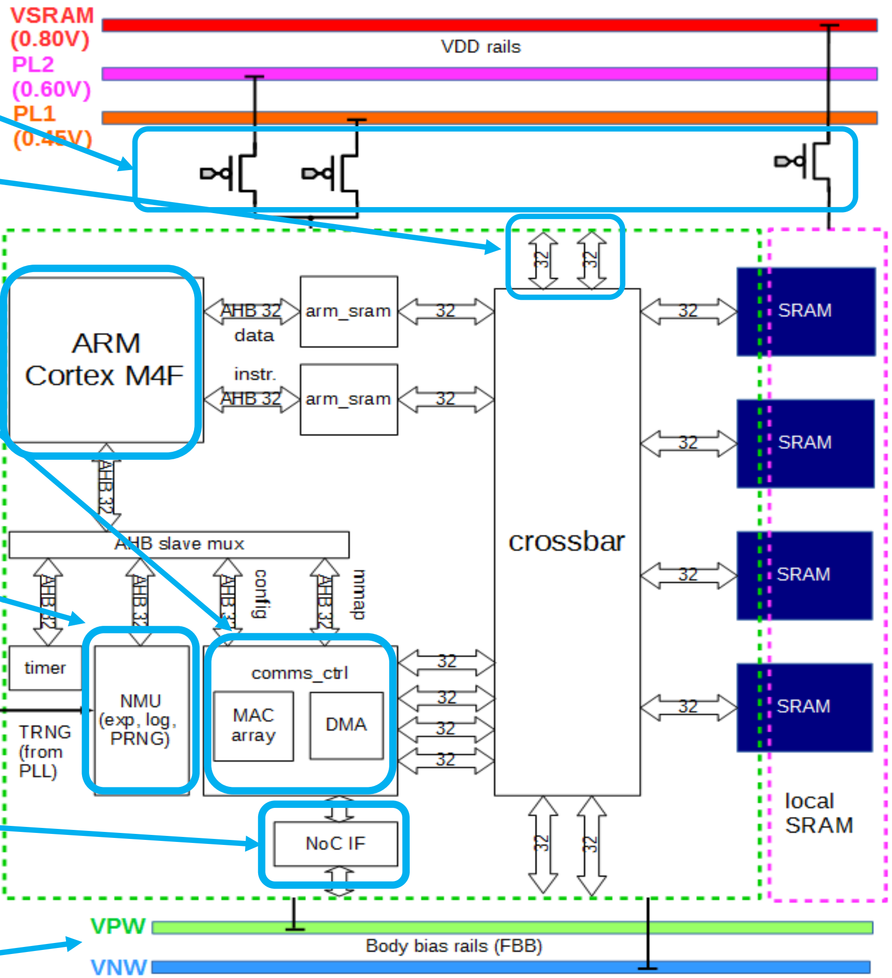
Memory sharing
- Synchronous access to neighbor PEs

Multiply-Accumulate accelerator
- MAC array with DMA 8-16 bits…

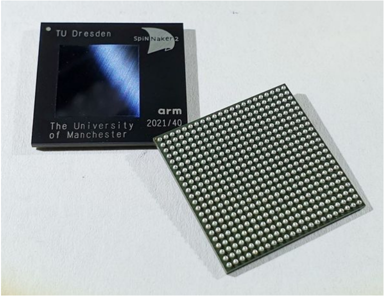
Neuromorphic accelerators
- Exp/log
- Random numbers (PRNG, TRNG from ADPLL noise)

Network-on-Chip
- On- and off-chip memory access
- SpiNNaker packet (spike) handling

Adaptive Body Biasing



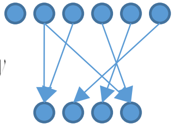
SpiNNaker2



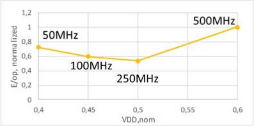
SpiNNaker2 Chip:
- 153 ARM cores
- >100 person design team
- 22FDX Global Foundries
- Developed in EU flagship Human Brain Project
- Development cost: >38Mio
- Deployment cost: >13Mio

Hybrid design for deep neural networks, spiking neural networks and symbolic AI

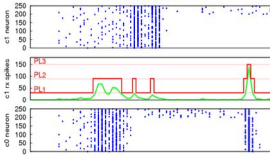
$$\tau_w \dot{w} = a(V_{Mem} - V_{rest}) - w$$



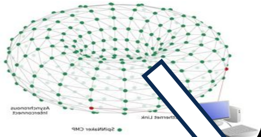
Outperforming Intel, Nvidia, Google on real time AI



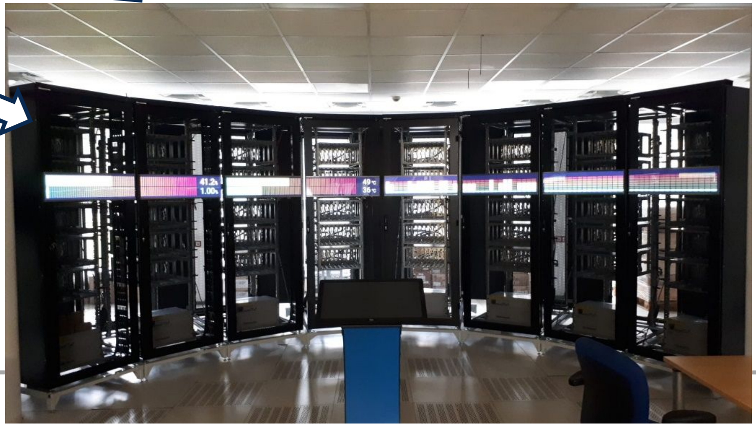
Brain-inspired **dynamic data sparsity**, i.e. ultra-efficient highly-parallel operation of AI algorithms on streaming data



Largest real-time AI platform worldwide, **10^14 parameters**
3 PFLOPS CPU
0.4 ExaOPS in AI accelerator



Physical: 10^7 processors, 70.000 chips, 14 racks, 10^{14} transistors



TECHNISCHE UNIVERSITÄT DRESDEN

Update 3: SpiNNaker2 Software

- **SNN simulation using PyNN**
  - Will re-use large parts from SpiNNaker1 stack (pyNN.sPyNNaker)
  - Current work: Adaption of low-level software
  - Availability: 2023 for 48-node boards, earlier for single-chip system
  - **Lava integration** -> BMBF project with Intel
- **DNN processing using Apache TVM**
  - Use TVM compiler to map large DNNs on SpiNNaker2 systems
  - Utilize machine learning accelerator for Conv2D, Dense and ReLU; other layer types supported by code generation
  - Can load DNNs trained in any common framework (TensorFlow, Pytorch, ...)
  - Status: SW devolopment started, examples on single chip expected in next half year
- **Hybrid SNN/DNN**
  - Light-weight Python interface for SNNs or hybrid networks on single chip
  - Available: now, already in use by 3 external groups
  - Serves a prototype for scalable Hybrid NN framework (combination of PyNN and TVM)

```
from spinnaker2 import snn, hardware

neuron_params = {
        "threshold":1.,
        "alpha_decay":0.9,
        }

stim = snn.Population(
        size=10,
        neuron_model="spike_list",
        params={0:[1,2,3], 5:[20,30]},
        name="stim")

pop1 = snn.Population(
        size=20,
        neuron_model="lif",
        params=neuron_params,
        name="pop1")
```

Still in development. Foundation with C++ on ARM.

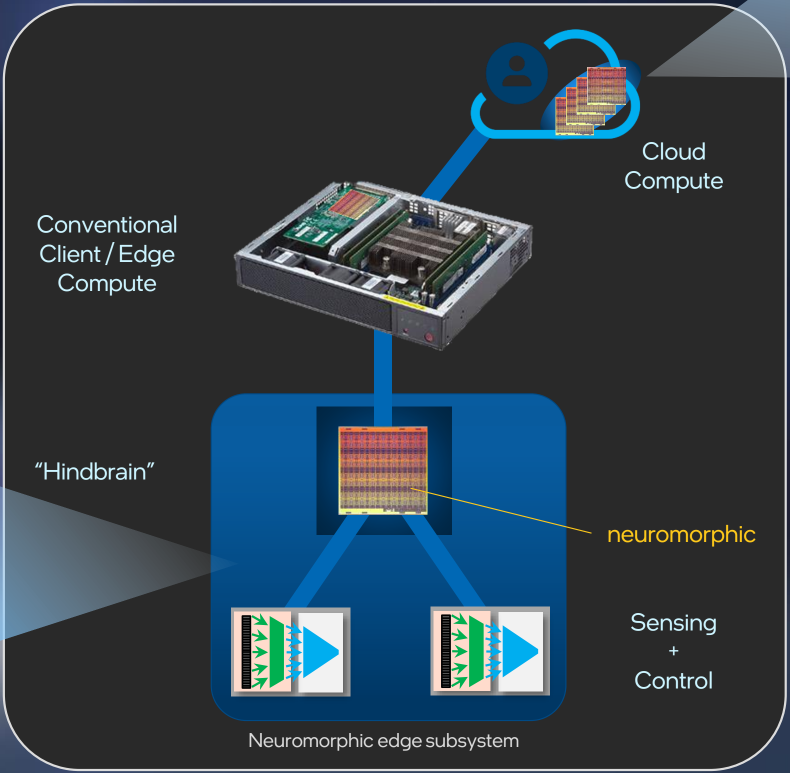
Loihi 2

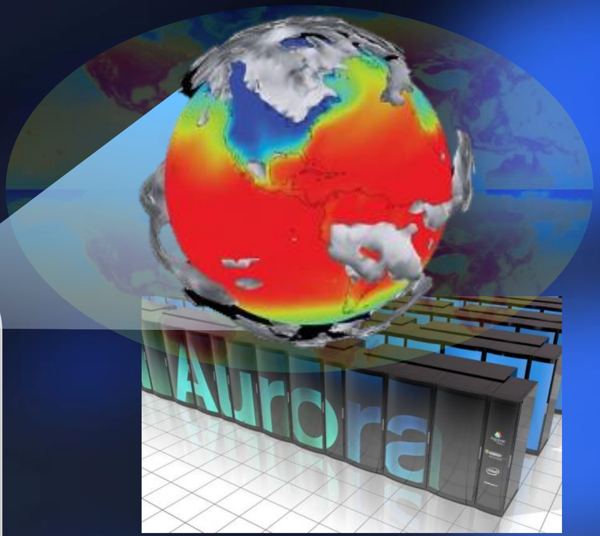
From INRC presentations

Research Vision

Integrate neuromorphic intelligence into computing products at all scales

Develop a new programmable computing technology inspired by the modern understanding of brain computation





Achieve brain-like efficiency, speed, adaptability, and intelligence

Deliver gains of 10^4 or higher in energy-delay-product*

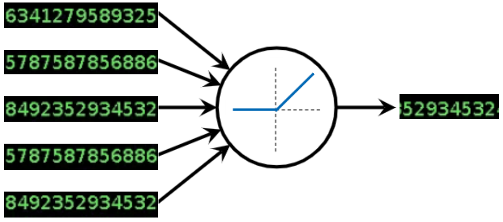
* Combined latency and energy efficiency metric

Exploiting dynamics at the neuron level

Maximize computation without data movement

Artificial Neuron (Stateless)

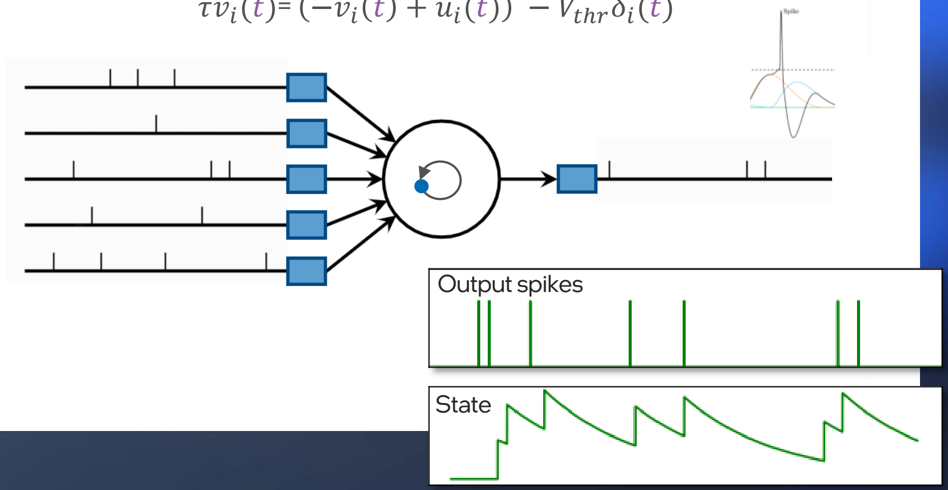
$$u_i = \sum_j w_{ij} f\left(u_j\right) + b_i$$



Spiking Neuron (Nonlinear Filter)

$$u_i(t) = \sum_j w_{ij}\left(\delta_j(t) * \alpha_u(t)\right) + b_i$$

$$\tau \dot{v}_i(t) = \left(-v_i(t) + u_i(t)\right) - V_{thr}\delta_i(t)$$



Realized in Loihi, improved in Loihi 2

KEY PROPERTIES

Compute and memory integrated to spatially embody programmed networks
Temporal neuron models (LIF) to exploit temporal correlation
Spike-based communication to exploit temporal sparsity
Sparse connectivity for efficient dataflow and scalability
On-chip learning without weight movement or data storage
Digital asynchronous implementation for power efficiency, scalability, and fast prototyping

Yet...

No floating-point numbers No multiply-accumulators No off-chip DRAM

Fundamental to deep learning hardware

Davies et al, "Loihi: A Neuromorphic Manycore Processor with On-Chip Learning." IEEE Micro, Jan/Feb 2018.

Challenges and Headwinds

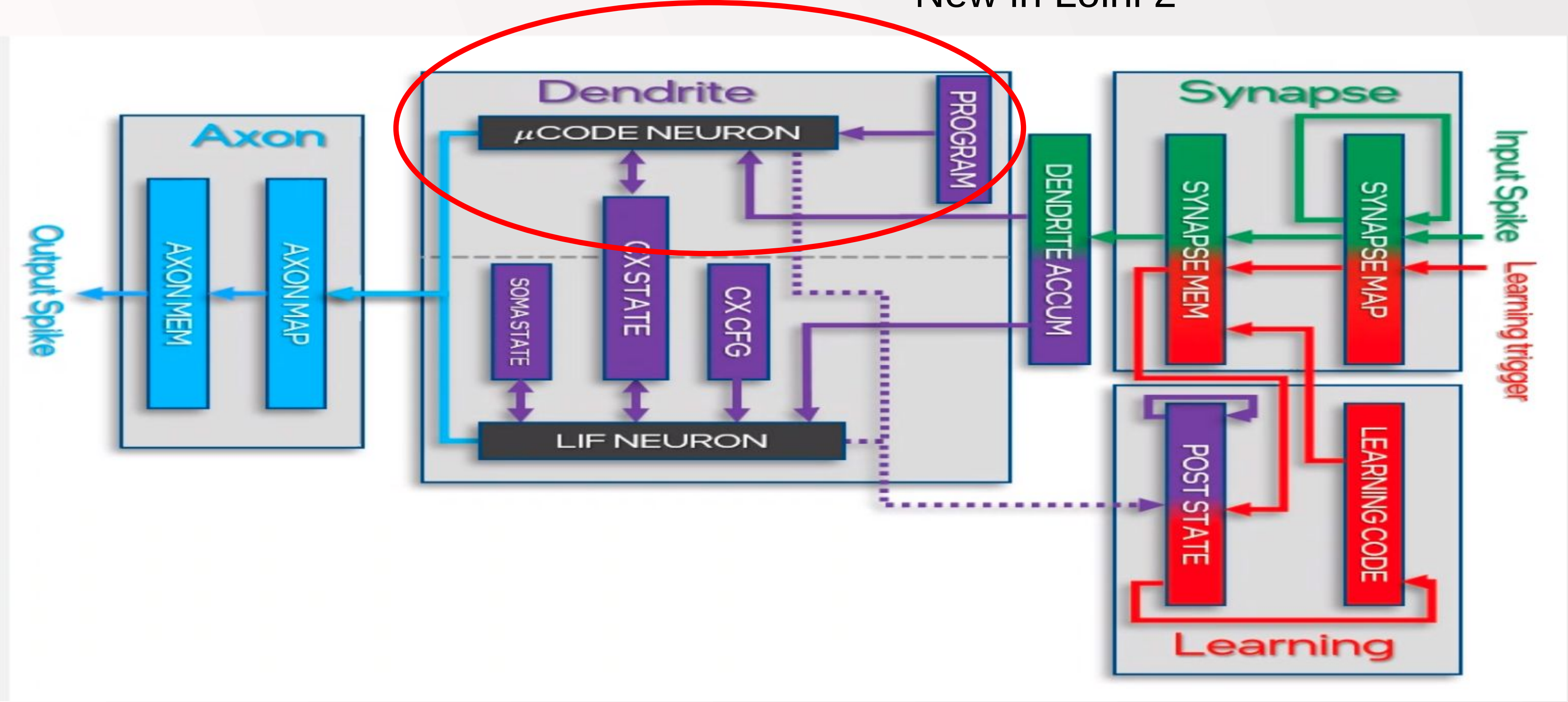
High cost due to on-chip memory integration

Algorithms and Programming models

Software convergence

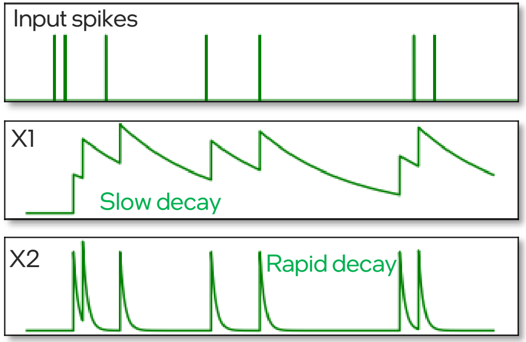
Loihi 2: Internal Neuron Model

New in Loihi 2

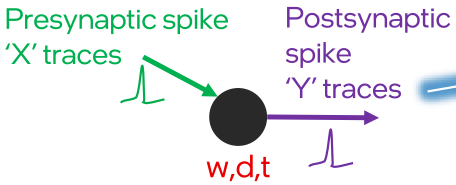


Enhanced synaptic plasticity for advanced online learning

Pre-synaptic Traces (X)
Input spikes exponentially filtered to generate pre-traces
Learning performs time-based pre-trace updates



Microcode Local Learning Rules
Synapse state updates using sum-of-product equations

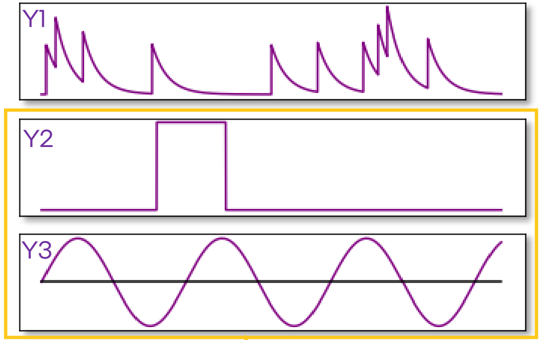


$$w' = w + \sum_{i=1}^{N_P} S_i \prod_{j=1}^{n_i} (V_{i,j} + C_{i,j})$$

Synaptic Variables
Wgt, Delay, Tag
(variable precision)

Variable Dependencies
X0, Y0, X1, Y1, X2, Y2,
Wgt, Delay, Tag, etc.

Post-Synaptic Traces (Y)
Loihi 1: LIF filters output spikes to generate traces

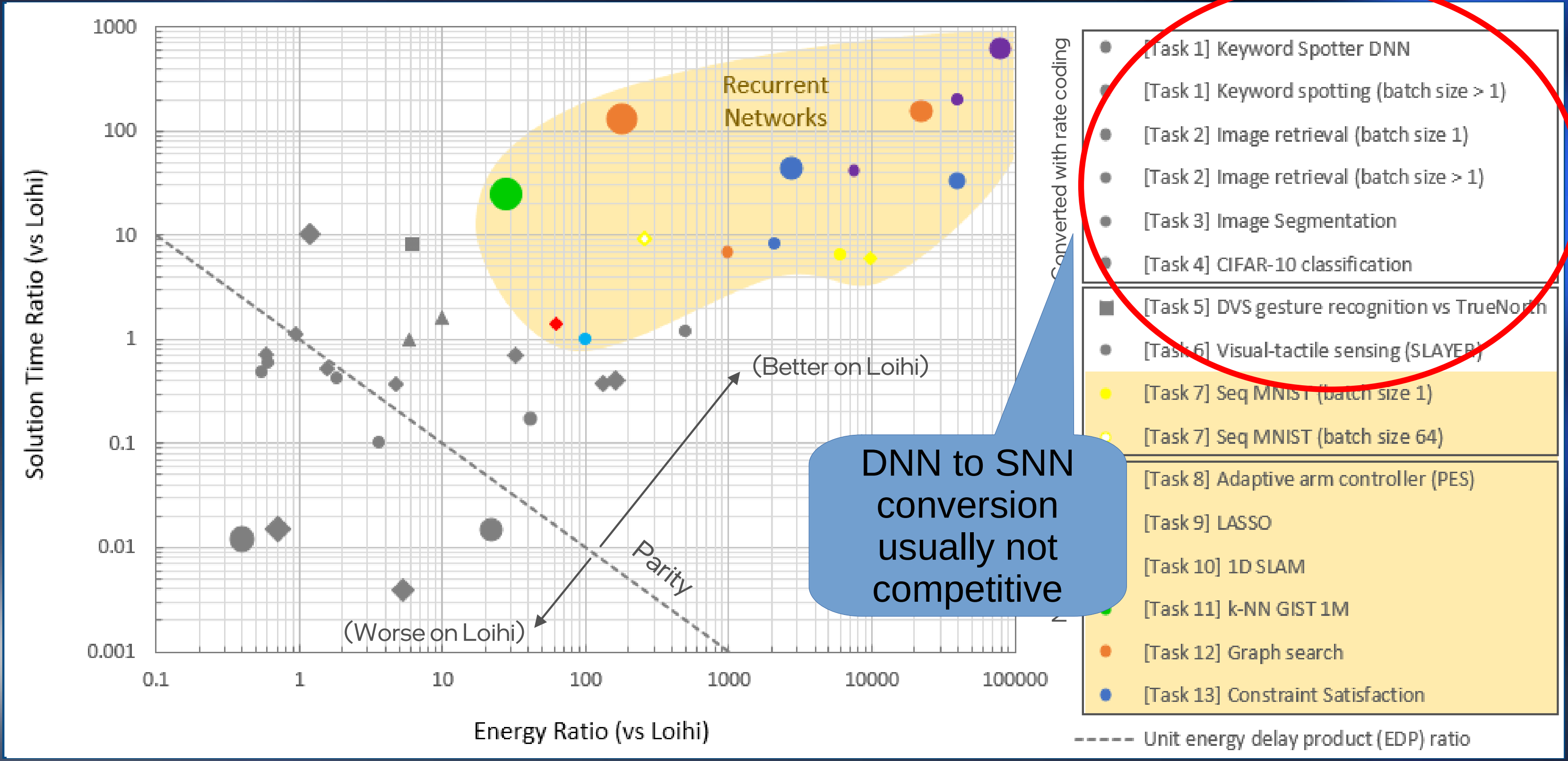


Loihi 2 neuron microcode can write arbitrary signed values to post-traces ("third factors")

Novel recurrent networks give the best gains

Reference architecture

- CPU (Intel Core/Xeon)
- GPU (Nvidia)
- Movidius (NCS)
- TrueNorth



M. Davies et al, "Advancing Neuromorphic Computing With Loihi: A Survey of Results and Outlook," Proc. IEEE, 2021. Results may vary.

Development platforms

- Lava https://lava-nc.org/ CPU, GPU and Loihi, plans for general neuromorphic
- Neural SNN simulators
  - NEST (python, C, GUI). No Loihi backend yet
  - Brian2 (python), with https://gitlab.com/brian2lava/brian2lava
  - PyNN (python). No backend yet
  - Nengo (GUI, proprietary scripting) https://www.nengo.ai/nengo-loihi/

https://www.intel.com/content/www/us/en/research/
neuromorphic-community.html
inrc_interest@intel.com

Developing Theory of NC computing

- Hyperdimensional computing/Vector Symbolic Architecture (HD/VSA): Gayler, Kanerva
- Computational graphs/GNN
  - POG, EPG in Zhang et al., Nature, 15.10.2020
  - SGNN, Yin et al. AAAI-24

Emerging Programming Paradigms

- Direct mapping, CPU neuron model → NC module
  - “ground truth” known from CPU, so many validation options.
- Optimization
  - Classical nonlinear, including DL with variants of grad descent (e.g. GDTT, surrogate gradient, eventprop)
  - Quantum optimizers (emulation)
  - Evolutionary Programming
- Continuous on-chip learning
  - Under research and development, some preliminary results

While ANN2SNN is a very efficient approach, the outcome is really suboptimal for SNN neuromorphic hardware (Loihi tests).

Emerging libraries

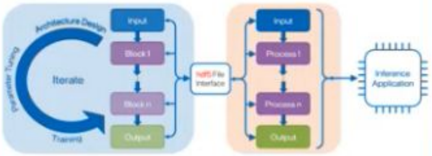
- Pre-trained modules
  - Edge processing
  - LSTM
- SNN transformers
  - Spikeformer; Event transformer
- LLM
  - SpikeGPT, SpikingBERT

For now training still off-line, on classical architectures.

Lava algorithm libraries

lava-dl

- Direct & HW-aware training of event-based DNNs
- Rich neuron model library (feed-forward & recurrent)

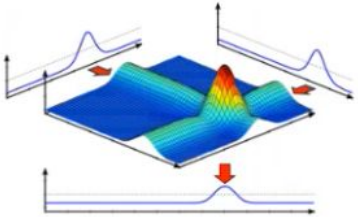


lava-optimization

- Family of constraint optimization solvers
- Today: QP, QUBO, LCA, BO
- Future: MPC, ILP, ...
- Standalone use or as part of AI applications

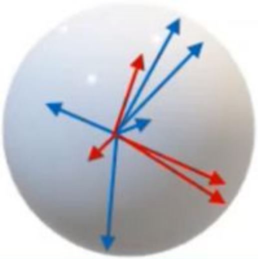
lava-dnf

- Design models with attractor dynamics
- Stabilize temporal data
- Selective data processing
- Dynamic working memories



lava-vsa (WIP)

- API for algebraic model description for VSAs
- Library of data types and operations (composition, binding, factorization, ...)



Constraints in developing for Loihi 2

- Algorithm level
  - Limited neural resources
    - Up to ~1 million neurons per chip
    - More restricted depending on topology and connectivity
  - Restricted topology of computational graphs (axon, synapse, neuron)
  - Best suited for sparse connectivity and data
- Process level
  - Access to local memory only
  - Fixed point arithmetic, limited precision (no floating point)
  - Limited instruction set
    - No division (although can be programmed)
    - No transcendental functions (e.g., logarithm, exponential, trigonometric)

But note that $x'(t) = -x(t)/\tau$ is the differential form of $\exp(-t/\tau)$

Advantages of SpiNNaker2 for non-AI Numerical Problems

Related projects on Loihi as well, w/o the quantum emulation.

1. Extreme parallelism of simple operations (think neurons…)
2. (Search for) Sparse solutions in high-dimensional numerical spaces
3. Stochastic computation/stochastic state representations
4. <u>Solving systems of locally coupled differential equations in a mesh/network topology (e.g. Neuron models, but also FEM and similar)</u>

	HPC	SpiNNaker2	Quantum Computing
Parallelism	10^5 cores	10^{14} synaptic updates/msec	$>10^{25}$ quantum entanglements
Stochastic Computation	Only in software, 10^{10} stochastic decisions/sec	Hardware accelerators, 10^{17} stochastic decisions/sec	Inherent in Qubits, $>10^{30}$ stochastic decisions/sec
Sparsity in high-dim spaces	Not supported	Fully supported	Fully supported
FEM-type tesselations	10^5 elements, boundary condition updates us to ms	10^7 elements in torus, boundary condition updates <10us	Potentially very fast convergence, but tessalation limited to #Qubits: 10^2-10^3

Conclusions

- Neuromorphic ecosystems are developing at a fast pace.
- Currently most rapid progress for digital neuromorphic
  - Good blend of performance/power and software support for Loihi, Akida and SpiNNaker2. Different tradeoffs in speed/energy/flexibility.
- Emerging workflows
  - Computational graphs, SGNN
  - Optimization
  - Physical simulations
  - Edge/robotics event-based AI
- Excellent review in *Neuromorphic hardware for sustainable AI data centers* https://doi.org/10.48550/arXiv.2402.02521